UNICAMP

UNIVERSIDADE ESTADUAL DE CAMPINAS
Faculdade de Engenharia Mecânica

WALLACE GUSMÃO FERREIRA

Efficient Global Optimization Driven by Ensemble of Metamodels: New Directions Opened by Least Squares Approximation

Otimização Eficiente Global Dirigida por Metamodelos Combinados: Novos Caminhos Abertos pela Aproximação por Mínimos Quadrados

CAMPINAS
2016

WALLACE GUSMÃO FERREIRA

Efficient Global Optimization Driven by Ensemble of Metamodels: New Directions Opened by Least Squares Approximation

Otimização Eficiente Global Dirigida por Metamodelos Combinados: Novos Caminhos Abertos pela Aproximação por Mínimos Quadrados

Thesis presented to the School
of Mechanical Engineering of the
University of Campinas, in partial
fulfillment of the requirements
for the degree of Doctor in
Mechanical Engineering, in the
area of Solid Mechanics and
Mechanical Design.

Tese de Doutorado apresentada
à Faculdade de Engenharia Mecânica
da Universidade Estadual de Campinas,
como parte dos requisitos exigidos
para a obtenção do título de
Doutor em Engenharia Mecânica,
na área de Mecânica dos Sólidos e
Projeto Mecânico.

Orientador/Supervisor: Prof. Dr. Alberto Luiz Serpa

CAMPINAS
2016

FICHA CATALOGRÁFICA ELABORADA PELA
BIBLIOTECA DA ÁREA DE ENGENHARIA - BAE - UNICAMP

Agência(s) de fomento e nº(s) de processo(s): Não se aplica.

Ficha catalográfica
Universidade Estadual de Campinas
Biblioteca da Área de Engenharia e Arquitetura
Luciana Pietrosanto Milla - CRB 8/8129

Ferreira, Wallace Gusmão, 1977-
F413e Efficient global optimization driven by ensemble of metamodels: : new directions opened by least squares approximation / Wallace Gusmão Ferreira. – Campinas, SP : [s.n.], 2016.

Orientador: Alberto Luiz Serpa.
Tese (doutorado) – Universidade Estadual de Campinas, Faculdade de Engenharia Mecânica.

1. Otimização. 2. Mínimos quadrados. 3. Superfícies de resposta. 4. Otimização global. 5. Otimização com restrições. I. Serpa, Alberto Luiz,1967-. II. Universidade Estadual de Campinas. Faculdade de Engenharia Mecânica. III. Título.

<u>Informações para Biblioteca Digital</u>

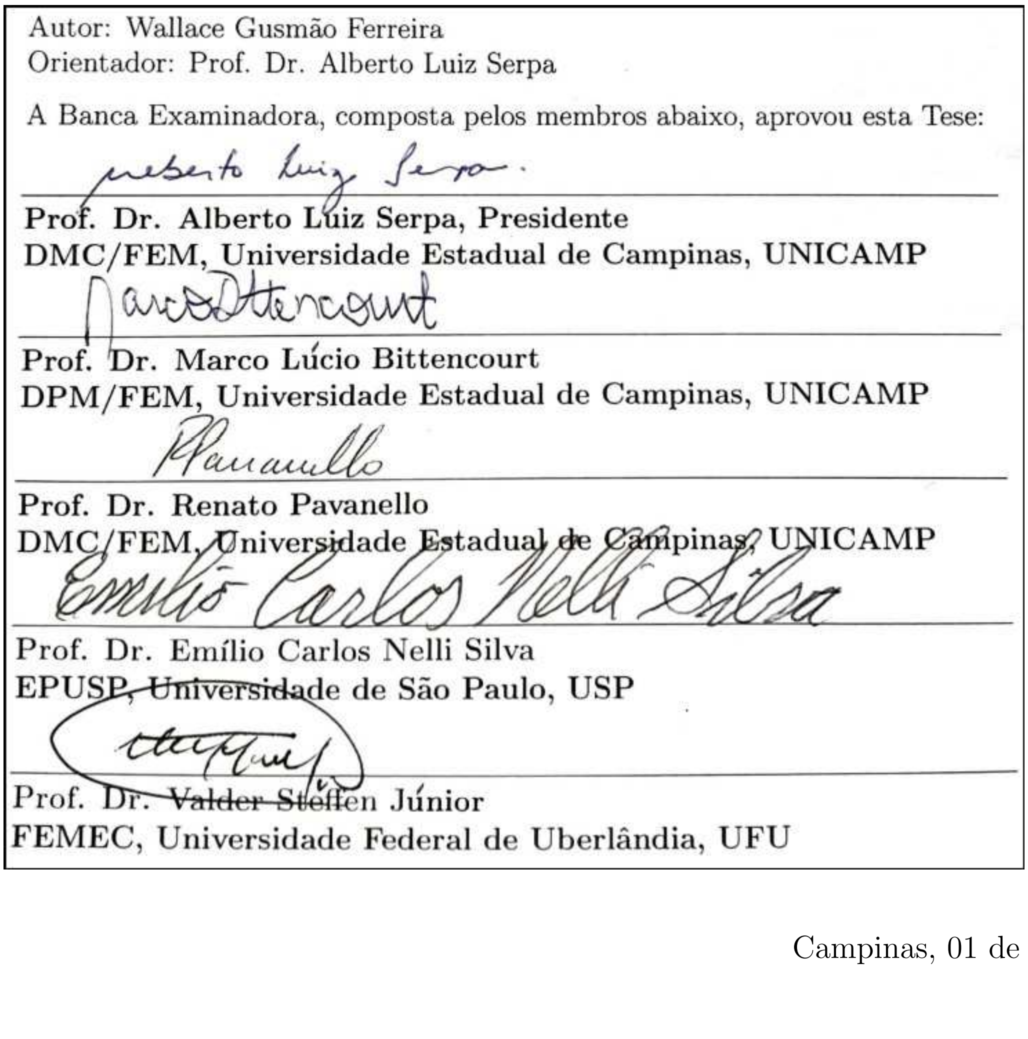
Título em outro idioma: Otimização eficiente global dirigida por metamodelos combinados : novos caminhos abertos pela aproximação por mínimos quadrados
Palavras-chave em inglês:
Optimization
Least squares
Response surfaces
Global optimization
Constrained optimization
Área de concentração: Mecânica dos Sólidos e Projeto Mecânico
Titulação: Doutor em Engenharia Mecânica
Banca examinadora:
Alberto Luiz Serpa [Orientador]
Emilio Carlos Nelli Silva
Marco Lúcio Bittencourt
Renato Pavanello
Valder Steffen Junior
Data de defesa: 01-12-2016
Programa de Pós-Graduação: Engenharia Mecânica

UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE ENGENHARIA MECÂNICA
COMISSÃO DE PÓ-GRADUAÇÃO EM ENGENHARIA MECÂNICA
DEPARTAMENTO DE MECÂNICA COMPUTACIONAL

TESE DE DOUTORADO ACADÊMICO

Efficient Global Optimization Driven by Ensemble of Metamodels: New Directions Opened by Least Squares Approximation

Otimização Eficiente Global Dirigida por Metamodelos Combinados: Novos Caminhos Abertos pela Aproximação por Mínimos Quadrados



Autor: Wallace Gusmão Ferreira
Orientador: Prof. Dr. Alberto Luiz Serpa

A Banca Examinadora, composta pelos membros abaixo, aprovou esta Tese:

[illegible]

Prof. Dr. Alberto Luiz Serpa, Presidente
DMC/FEM, Universidade Estadual de Campinas, UNICAMP

[illegible]

Prof. Dr. Marco Lúcio Bittencourt
DPM/FEM, Universidade Estadual de Campinas, UNICAMP

[illegible]

Prof. Dr. Renato Pavanello
DMC/FEM, Universidade Estadual de Campinas, UNICAMP

Emilio Carlos Nelli Silva

Prof. Dr. Emílio Carlos Nelli Silva
EPUSP, Universidade de São Paulo, USP

[illegible]

Prof. Dr. Valder Steffen Júnior
FEMEC, Universidade Federal de Uberlândia, UFU

Campinas, 01 de Dezembro de 2016.

"If I had more time,

I would have written a shorter letter..."

Blaise Pascal, 1623-1662.

"Não é o conhecimento,

mas o ato de aprender,

não é a posse,

mas chegar lá, que promove

o maior encantamento."

J. C. F. Gauss, 1777-1855.

Dedicatória

Este trabalho é dedicado a
Patrícia Vendramim,
minha Musa Inspiradora.
E aos pequenos Heitor e Ícaro,
nossas Obras Primas.

Sem vocês na minha vida,
nada mais importa!

Agradecimentos

Há certas coisas que começamos e não terminamos. Outras, nem chegamos a tentar. Este trabalho começou em algum lugar do passado e, depois de uma caminhada longa, não tenho mais clareza sobre quando realmente comecei, nem tampouco porque comecei... O fato é que, mesmo nas horas de maior desmotivação, sempre tive a certeza de que iria terminar algum dia, ou pelo menos iria morrer tentando...

Hoje, depois de muitas tentativas, erros e sucessos, horas de sono perdidas, dezenas de artigos e capítulos lidos (e outras dezenas que ficaram somente na promessa para ler algum dia), rugas, alguns cabelos já esbranquiçados, entre outras venturas e desventuras, eis que dou-me por satisfeito e é chegada a hora de colocar um ponto final nessa estória e iniciar outros projetos. Não sei bem se terminei, mas tenho plena certeza de que percorrer esse caminho valeu muito a pena!

Durante essa longa jornada pude contar com o Prof. Alberto Serpa, que me aceitou como aluno-orientado em tempo parcial. Sempre com muita paciência e cortesia, tolerou todos os meus atrasos, dúvidas, mudanças de tema de pesquisa e também sempre tentou me manter motivado a chegar no fim da viagem. Seu amplo conhecimento e experiência acadêmica e industrial, sua visão clara e objetiva e conselhos precisos foram de grande utilidade e me guiaram até aqui. Além do mais, sempre leu prontamente, no mínimo detalhe, todos os meus manuscritos, os quais culminaram nesse texto de tese. Entre outras coisas, aprendi também que "a vida deve ser encarada como uma maratona e não uma prova de cem metros rasos".

Agradeço aos professores membros da banca do exame de qualificação e defesa desse trabalho e também aos revisores e editores para os quais submetemos os manuscritos para publicação. Todas as correções e sugestões são de muita valia, contribuindo sobremaneira para o texto chegar a esse ponto.

A infraestrutura para o desenvolvimento desse trabalho foi toda baseada na implementação da SURROGATES Toolbox, compilada e desenvolvida pelo Dr. Felipe A. C. Viana, o qual, além de deixar o código aberto e em domínio público, sempre foi muito gentil e prestativo a sanar dúvidas e além do mais contribuiu com comentários e sugestões pertinentes no início dessa pesquisa. Mais do que isso, grande parte do trabalho desenvolvido nessa tese tem como principal referêcia os artigos publicados por Dr. Viana e seus colaboradores.

Sem dúvida este trabalho não teria sido possível sem o apoio da Ford Motor Company e pela colaboração de vários colegas e gestores. Foram de grande importância principalmente os meus supervisores diretos ao longo desse tempo: Gustavo Conrado (MSX), Luis Coutinho (MSX), João S. Filho, Fernando Mendonça, José Felício, Ricardo Albuquerque, Ricardo Cerqueira e Ciro Almeida. Obrigado por permitirem que eu dedicasse parte das minhas horas de trabalho para desenvolver essa pesquisa. Sem esquecer também da imensa base de dados de pesquisa científica RLIS (Research Library and Information System), mantida pela Ford, sem a qual eu provavelmente não conseguiria acesso a muitos livros e artigos, de forma totalmente gratuita e sem burocracia.

Ainda com relação à Ford, um agradecimento especial ao Paulo Alves e ao Marcelo Magalhães por terem me dado a oportunidade de fazer parte do time de MDO (Multidisciplinary Design Optimization) e passar um período de trabalho e treinamento em Dearborn, Michigan (EUA). Essa experiência foi de grande importância pessoal e profissional e pude trocar conhecimentos nesse tempo com várias pessoas, entre elas, Rodrigo Miyamoto, Rairam Almeida, Giri Namalwar, Aditya Raghunathan, Yogita Pai, Ariel Chintha, Chin-Hung Chuang, Joana Rakowska, Amir Chator e aos colegas da Ford Alemanha: Michel Paas, Hessel van Dijk e Thomas Erber. Esse período me fez aprender muito sobre métodos de otimização multidisciplinar e metamodelagem, além de ter feito grandes amizades!

Gostaria de registrar também um agradecimento à Altair Engineering do Brasil e toda a sua equipe, em especial: Valdir Cardoso, Yula Massuda, Karen Silva, Adriano Koga e Rogerio Nakano, sempre solícitos e prestativos no suporte técnico e outras "ajudas", mais especificamente com o pacote HyperWorks, utilizado em parte dos estudos de caso dessa tese.

Como não poderia faltar, meus pais e irmãos, por sempre me apoiaram e me incentivaram a perserverar e a me desenvolver através dos estudos. A vocês: Romeu, Guida, Jane, Vanessa, Júnior e Carla, sem me esquecer de Diva e Ricardo, meu muito obrigado!

Nos vários momentos em que me isolei para desenvolver esta pesquisa e escrever os textos da tese, sacrifiquei horas importantes da convivência e companhia da minha amada esposa Patrícia e dos nossos dois pequenos filhos Heitor e Ícaro. Espero poder recompensá-los a contento! Obrigado por todo o apoio e paciência e por suportarem todos os momentos em que eu fiquei longe de vocês.

Esse trabalho teve a ajuda ou influência, torcida ou cobrança, de várias pessoas, entre amigos, colegas de trabalho e familiares. Listar todos não seria possível, não haveria espaço e, com certeza, eu me esqueceria de alguém. Como disse o professor R. S. Waslawick, não vou mencionar os nomes pois “eles já sabem quem são”. Dessa forma, a todos vocês, muito obrigado!

São Bernardo do Campo, SP, Brasil, Dezembro de 2016.

Resumo

FERREIRA, Wallace Gusmão. *Otimização Eficiente Global Dirigida por Metamodelos Combinados: Novos Caminhos Abertos pela Aproximação por Mínimos Quadrados.* Faculdade de Engenharia Mecânica, Universidade Estadual de Campinas. Campinas, SP, Brasil, 2016. 250p. (Tese de Doutorado, em Inglês)

O presente trabalho representa a compilação dos resultados anteriores dessa pesquisa no campo de metamodelos combinados e otimização eficiente global (EGO), os quais foram submetidos para publicação em periódicos especializados. Recentemente foi implementado nesse trabalho de doutorado o algoritmo LSEGO que é uma abordagem para conduzir algoritmos tipo EGO, baseando-se em metamodelos combinados através da aproximação por mínimos quadrados (metamodelos combinados LS). Através dos metamodelos combinados LS é possível estimar a incerteza da aproximação usando qualquer tipo de metamodelagem (e não somente do tipo kriging), permitindo estimar a função de expectativa de melhora para a função objetivo. Nos experimentos computacionais anteriores em problemas de otimização sem restrições, a abordagem LSEGO mostrou-se como uma alternativa viável para conduzir otimização eficiente global usando metamodelos combinados, sem se restringir a somente um ponto adicional por ciclo de otimização iterativa. Na presente tese, o algoritmo LSEGO foi extendido de modo a tratar também problemas de otimização com restrições. Os resultados de testes numéricos com problemas analíticos e de referência e também em um estudo de caso de engenharia em escala industrial mostraram-se bastante promissores e competitivos em relação aos trabalhos similares encontrados na literatura.

Palavras Chave

- Otimização via metamodelos, Metamodelos combinados, Mínimos quadrados, Otimização eficiente global , Otimização com restrições

Abstract

FERREIRA, Wallace Gusmão. *Efficient Global Optimization Driven by Ensemble of Metamodels: New Directions Opened by Least Squares Approximation.* School of Mechanical Engineering, University of Campinas - UNICAMP. Campinas, SP, Brazil, 2016. 250p. (Doctor's Thesis, in English)

In this work we review and compile the results of our previous research in the fields of ensemble of metamodels and efficient global optimization (EGO). Recently we implemented LSEGO that is an approach to drive EGO algorithms, based on LS (least squares) ensemble of metamodels. By means of LS ensemble of metamodels, it is possible to estimate the uncertainty of the prediction by using any kind of model (not only kriging) and provide an estimate for the expected improvement function. In previous numerical experiments with unconstrained optimization problems, LSEGO approach has shown to be a feasible alternative to drive efficient global optimization by using multiple or ensemble of metamodels, not restricted to kriging approximation or single infill point per optimization cycles. In the present work we extended the previous LSEGO algorithm to handle constrained optimization problems as well. Some numerical experiments were performed with analytical benchmark functions and also for industry scale engineering problems with competitive results.

Keywords

- Surrogate based optimization, Ensemble of metamodels, Least squares, Efficient global optimization, Constrained optimization

Contents

1 Introduction **15**

1.1 Technological Scenario . . . 15

1.2 Thesis Subjects and Research Scope . . . 17

1.3 Objectives and Original Contributions of the Thesis . . . 22

1.3.1 Phase I . . . 23

1.3.2 Phase II . . . 23

1.3.3 Phase III . . . 24

1.3.4 Summary of Implementations . . . 25

1.4 Thesis Chapters Outline . . . 25

2 Literature Overview **29**

2.1 Some Historical Remarks . . . 29

2.2 Specific Literature for the Thesis Scope . . . 35

2.2.1 Metamodel Based Optimization . . . 37

2.2.2 Efficient Global Optimization . . . 38

2.2.3 Ensemble of Metamodels and EGO . . . 40

3 Metamodel Based Analysis and Optimization **43**

3.1 Metamodel Creation Methods . . . 45

3.2 Metrics for Metamodel Validation . . . 46

3.3 Metamodel Based Optimization . . . 47

3.4 An Illustration with Metamodels . . . 49

4 Ensemble of Metamodels Background **51**

4.1 Origins and Definitions . . . 51

4.2 An Example with Simple Average Ensemble . . . 54

4.3 Ensemble Methods in Engineering Design . . . 55

5 Ensemble of Metamodels by Least Squares **58**

5.1 Basic Formulation . . . 58

5.2 Multicollinearity and Least Squares . . . 62

5.3 The Augmented Least Squares Approach . . . 64

5.4 An Example with Least Squares Variants . . . 67

5.5 The LS Ensemble Approach: Results Summary . . . 71

6 Efficient Global Optimization (EGO) **74**

6.1 Basic Formulation of EGO . . . 75

6.2 The Standard EGO Algorithm . . . 76

6.3 On Balancing Exploration and Exploitation . . . 77

6.4 Extensions for the EGO algorithm . . . 79

7 LS Ensemble of Metamodels EGO (LSEGO) 81
7.1 LSEGO Algorithm with Parallel Infill Points 85
7.2 Illustrations: One and Two Variables Examples 86
7.3 The LSEGO Approach: Results Summary 95

8 Applications to Constrained Optimization Problems 99
8.1 Computer Implementation 100
8.1.1 Ensembles of Metamodels 100
8.1.2 Maximization of the Expected Improvement Functions 101
8.2 Test Problems 102
8.2.1 Analytical Benchmark Functions 102
8.2.2 Analytical Engineering Problems 102
8.2.3 Engineering Application: Car Impact 106
8.3 Design of Experiments (DOE) 109

9 Results and discussion 110
9.1 Analytical Benchmark Functions 110
9.1.1 Braninh-Hoo 110
9.1.2 Hartman-3 111
9.1.3 Hartman-6 111
9.1.4 Summary for Analytical Benchmarks 115
9.2 Analytical Engineering Problems 117
9.2.1 Three-Bar Truss 117
9.2.2 Cantilever Beam 122
9.2.3 Helical Spring 124
9.2.4 Pressure Vessel 125
9.2.5 Sequential Sampling vs. One-Stage Approach 128
9.2.6 Unfeasible Results Near Constraints Boundaries 129
9.3 Engineering Computer Model: Car Impact 131

10 Concluding Remarks 134
10.1 Summary of Thesis Original Contributions 134
10.2 Overall Achievements 135
10.3 Some Possible Future Directions 136

References 138

A SURROGATES Toolbox 147

B Boxplots Definition 148

C The Kriging Metamodel 149

D Test Functions 150

D.1 Analytical Benchmarks 150
D.2 Engineering Applications 152

E Preliminary Numerical Study 155
E.1 Introductory Note 155
E.2 Abstract 155
E.3 Numerical experiments 155
E.3.1 Tests with an analytical function 156
E.3.2 Tests with engineering problems 157
E.4 Results and discussion 159
E.4.1 Results with the analytical function 159
E.4.2 Results with engineering problems 164

F Multicollinearity and Least Squares 169
F.1 The Sources of Multicollinearity 169
F.2 Prior Model Selection 170
F.3 Gathering Additional Data 172
F.4 Variable Selection Methods in Regression 172
F.4.1 The Bias and Variance Dilemma 172
F.5 Least Squares Variants 175
F.5.1 Stepwise Regression 175
F.5.2 Ridge Regression 178
F.5.3 Principal Component Regression 180
F.5.4 Penalized Least Squares: A Unified Approach 184
F.6 Other Least Squares Variants to Improve Accuracy 187
F.6.1 Effect of the Intercept Term 187
F.6.2 Generalized Least Squares 188
F.6.3 L_p-Norms and Robust Regression 191
F.6.4 Total Least Squares 197

G Manuscripts Submitted 202

1 Introduction

"Begin at the beginning and go on till you come to the end: then stop."

Alice in the Wonderland

Lewis Carrol, 1832-1898.

1.1 Technological Scenario

Structural Optimization is a recognized and matured research area with more than thirty years of increasingly intensive work worldwide. Extensive reviews can be found on the area of optimization and engineering design, for instance: Sobieszczanski-Sobieski and Haftka (1997), Haftka et al (1998), Papalambros (2002), Saitou (2005), Roy et al (2008), Yang et al (2013b) and in the recent book Arora (2012).

As discussed by several authors in the technical literature, one of the main obstacles for the large use and acceptance of structural optimization methods since its beginning is the *computer power*. In the mid 1970s, the beginning of computer based structural optimization, the computation time required for running analysis was the major drawback due to the limited computer resources available.

On the other hand, the last twenty years were marked by the widespread availability of low-cost and high-performance computers (or clusters) and also powerful commercial simulation software (e.g., finite elements, FEM; multibody dynamics, MBD; and computational fluid dynamics, CFD). This availability of computational resources leveraged the integration of structural analysis and numerical optimization methods applied in practical engineering development, instead of only in research or academic laboratories.

Paradoxically, even with the significant increase in hardware and software resources in the last years, computation power is still a problem. Besides the high computation power available today, the demands in terms of higher accuracy, complex analysis (e.g., nonlinear, transient, multiphysics, multidisciplinary), and large design domains are continuously and rapidly increasing. The dilemma looks like to be "infinite", i.e., the more computational power we have, the more complex are the problems we want to solve. This "never-ending" need of computer resources in structural optimization is discussed in an interesting essay by Venkatararaman and Haftka (2004).

As an example of this kind of situation faced by the engineers, the typical multidisciplinary design and optimization (MDO) problem found nowadays at automotive industry is presented in Fig. 1. If, for instance, it is desired to optimize the vehicle mass (or cost), by considering several disciplines, i.e., Safety, Durability and NVH (noise, vibration and harshness), the problem should involve hundreds of design parameters (e.g., sheet metal thickness and material properties) and hundreds of functional responses to be monitored (e.g., stress limits, natural frequencies, displacements, accelerations, etc).

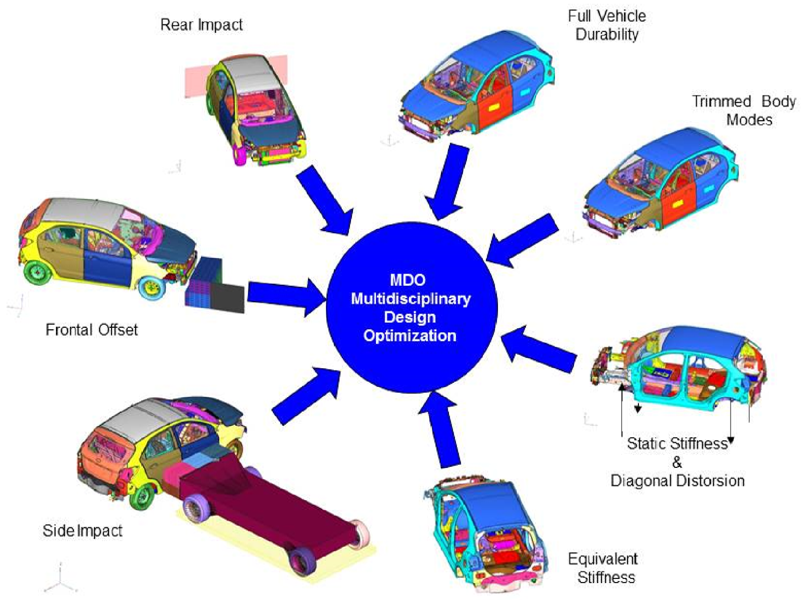


Figure 1: Typical multidisciplinary design and optimization (MDO) problem at automotive industry. Courtesy of Ford Motor Company. Reproduced with permission.

For example, even with high-end computers clusters used nowadays in automotive industry, one single full vehicle analysis of high fidelity safety crash, by finite elements, takes up to 12 processing hours with 48 CPU in parallel. By its turn, one single full vehicle aerodynamics analysis for drag calculation, using 96 CPU, should take up to 30 processing hours to finish.

In this context, it is virtually impossible to drive sensitivity analysis and/or multidisciplinary optimization by using *directly* the original computer models (e.g., FEM or CFD models) to evaluate hundreds or thousands of designs proposals in feasible engineering time. Henceforth, another alternative approach must be applied, with lower computational cost and also acceptable accuracy, in a way to make the optimization process feasible.

In this sense, the approximation methods for computer models (also known as *metamodels*, *surrogate models*, *response surfaces*) are considered key technologies to provide or improve the feasibility to solve complex engineering problems as described in Fig. 1, for instance.

In summary, surrogate based (or response surface based, or metamodel based) optimization[1] refers to the process of using approximate objective and constraint functions to drive the standard optimization algorithms available. In other words, instead of using the original *time consuming* computer simulation models directly in the optimization process, they are replaced by *surrogate functions*, or metamodels that can be *accurate* and *fast running* at same time to foster the feasibility of the whole optimization process.

This methodology, known as *metamodel based analysis and optimization*, has shown to be effective in multidisciplinary/multiobjective optimization scenarios and it has been widely applied in both research and industry, for example automotive, aerospace and oil-gas prospection.

In this thesis we will present our developments and results in the field of metamodel based analysis and optimization. The overall process can be roughly split in two main research fronts: (i) the creation of accurate metamodels, with limited time and computer resources and (ii) the management of the whole metamodel based optimization process in an efficient way.

In the next section we will provide more details regarding the main research subjects and literature references needed to develop this thesis work.

1.2 Thesis Subjects and Research Scope

Roughly speaking this research is devoted to a main branch in the **Structural Optimization** field known as **Metamodel Based Optimization**. Four our purposes here, this branch of research can be divided in two other branches, i.e., **Ensemble of Metamodels** and **Efficient Global Optimization**.

In the present section it is presented only a brief description of these research subjects and the relevant references in the technical literature that inspired the development of the present thesis work. Further and detailed discussion will be presented in the Literature Overview (Chapter 2) and in the respective chapters along this thesis text.

As we mentioned in the previous section, our main research topic is the metamodel based analysis and optimization processes and methods. See in Fig. 2 the comparison of two main approaches available to treat complex optimization problems as displayed in Fig. 1, for example.

[1]Metamodel in simple terms means “model of a model” and its first use is acknowledged to Kleijnen (1985). The terms *response surface*, *surrogate* or *metamodel* have the same meaning in the context presented here. We will use metamodel more frequently (and sometimes surrogate) since they are the forms commonly found in the engineering optimization literature.

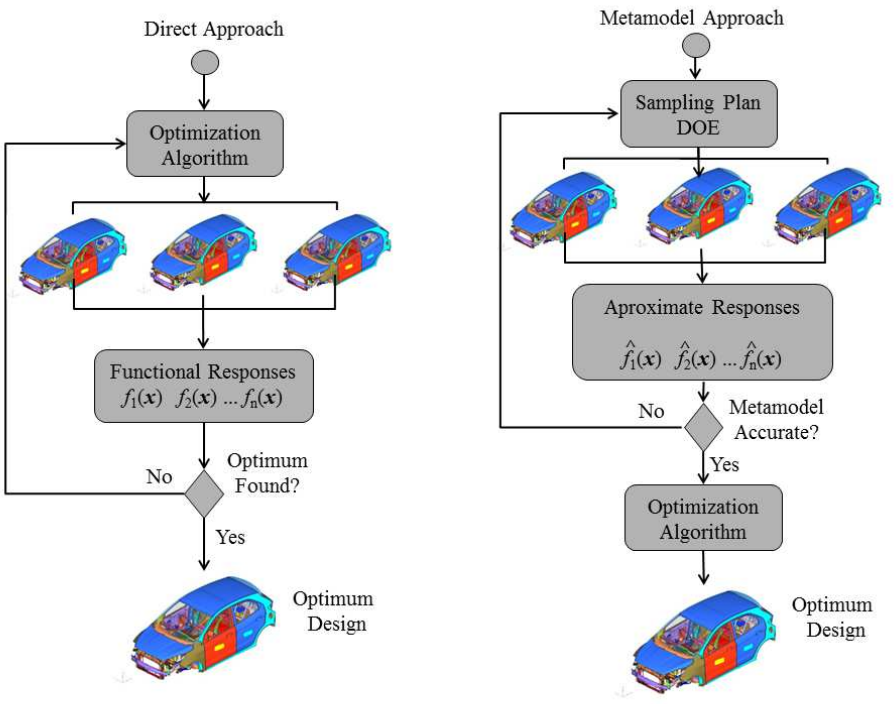


Figure 2: *Direct Optimization* (left) versus *Metamodel Based Optimization* (right).

In the *direct* approach, Fig. 2 (left), at each iteration or cycle, the optimization algorithm submits the simulation models to run in order to evaluate the performance responses $f_i(\mathbf{x})$. This process can be repeated for several iterations, with hundreds or thousands function evaluations.

On the other hand, in Fig. 2 (right), the *metamodel based* approach (also called *surrogate based* or *response surface based*), a design of experiments (DOE) is created and the simulation models are submitted to evaluate the performance responses $f_i(\mathbf{x})$. This process is repeated until reasonable approximations $\hat{f}_i(\mathbf{x})$ for the responses are achieved. At the end, the optimization algorithm uses the validated metamodels $\hat{f}_i(\mathbf{x})$ in the search of the optimum design.

In practical terms, the direct approach is prohibitive (or even impossible) to be applied in MDO problems as in Fig.1. On the other hand, the metamodel approach is feasible and efficient in such complex MDO scenarios.

In summary, metamodel based optimization refers to the process of using fast running metamodels, as surrogates of original complex and long time running computer simulation models, with the aim to approximate the *true* objectives and constraint functions to be used inside a standard optimization algorithm.

Metamodel Based Optimization is therefore the main branch of our research and it has been widely applied in research and industry. Refer to Queipo et al (2005), Simpson et al (2008) and Forrester and Keane (2009) for broader discussion on this subject.

Nowadays it is known that the best metamodeling strategy is always dependent on: the degree of nonlinearity of the problem; the number of design variables; the sampling points technique and distribution in the design space; the functional form of surrogate model (shape and tuning parameters) and the error measure adopted.

After a lot of research and comparative studies in this subject performed in the last decades, the conclusion is that it is a problem dependent issue. There is not a unique approximation method that can handle with equal and acceptable accuracy every problem at hand, or at least the majority of problems to approximate. Each method has its pros and cons and the user should be able to choose the best for his own purposes.

The task for selecting metamodels has not a trivial solution and, since the computational cost for creating several metamodels is small as compared to the cost to evaluate the true function being approximated, and henceforth the combining process or a mixed "selecting and combining" approach can be a feasible way.

In this sense, concepts like *ensemble of predictors*, *mixture of experts*, *committee of networks*, etc., are well known in Machine Learning literature, see for instance Wolpert (1992), Bishop (1995), Breiman (1996) and more recently Seni and Elder (2010). The idea is that, by the combination of different predictors, it is possible to improve the final model accuracy for both functional evaluation or pattern recognition purposes.

Motivated by the potential benefits, the research branch of **Ensemble of Metamodels** methods has gained attention in the last years in the structural optimization engineering field as well. For functional evaluation in general the ensemble of metamodels is formed as a linear combination (weighted sum) of different models in a set. For this reason, the ensemble of metamodels are also known as WAS (weighted averaged surrogates). This is an active area of research that is getting maturity, as can be seen in the publications on this topic in the last years: Viana et al (2009), Seni and Elder (2010), Viana (2011), Zhou (2012), Zhang and Ma (2012) and Yang et al (2013b). The first part of our research is concentrated in this branch and we will present the details in the next section.

Another branch of research is the so called **Efficient Global Optimization** (EGO), that is a type of surrogate/metamodel based optimization approach with iterative (or sequential) sampling. In this kind of approach, the metamodels $\hat{f}_i(\mathbf{x})$ are created and the expected improvement functions $E_i[I_i(\mathbf{x})]$ are maximized to generate additional points in the sampling space (*infill points*) to be used in the next optimization cycle. The process iterates and the *global*[2] optimum is found when the level of expected improvement is negligible or the maximum allowed number of cycles/iterations is reached. See in Fig. 3 a basic flowchart for the EGO approach.

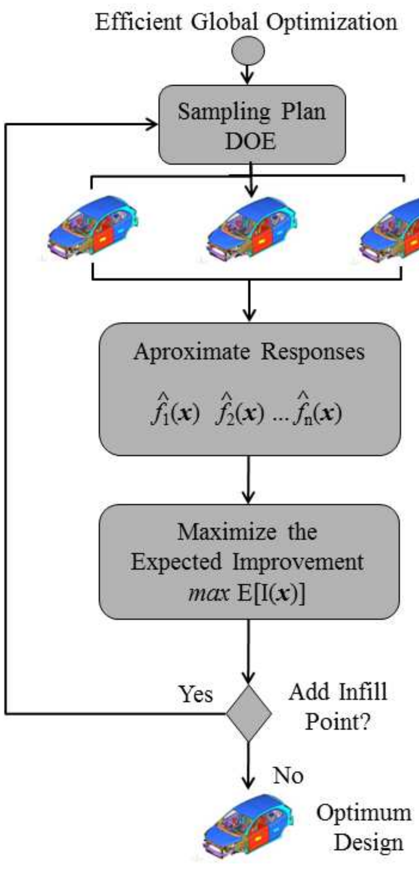

Efficient Global Optimization
Sampling Plan DOE
Aproximate Responses
$\hat{f}_1(\boldsymbol{x})\ \hat{f}_2(\boldsymbol{x}) \ldots \hat{f}_n(\boldsymbol{x})$
Maximize the Expected Improvement
max E[I($\boldsymbol{x}$)]
Yes
Add Infill Point?
No
Optimum Design

Figure 3: Efficient global optimization (EGO) process overview.

The EGO concept emerged after the work of Jones et al (1998), that was based mainly on previous research on Bayesian Global Optimization (Schonlau (1997) and Mockus (1994)). Traditionally the available EGO algorithms are based on the kriging metamodel approach and a single infill point is generated per optimization cycle.

It can be observed an increasing research interest on EGO approaches with multiple infill points per cycle published in the last years. See for example the works by Sóbester et al (2004), Henkenjohann and Kukert (2007), Ponweiser et al (2008), Ginsbourger et al (2010) and Viana et al (2013). Other recent EGO developments can be found in Rehman et al (2014), Mehari et al (2015) and Encisoa and Branke (2015).

In Viana et al (2013), it was suggested an alternative scheme to generate multiple (or

[2]The term *global optimization* is used in this thesis by following the historical nomenclature for this family of methods. The proper term should be "best local optimum" in the design space, since the domain is restrict by definition in the metamodel based optimization and thus the "true global optimum" is not guaranteed to be found.

parallel) infill points per cycle in the EGO algorithm by taking advantage of multiple surrogates, or ensemble of metamodels. In this direction, as we will detail in the next section, the second part of our research is concentrated on the combination of the advantages of **Ensemble of Metamodels** and **Efficient Global Optimization**.

In summary, the focus of this thesis can be better explained with the flowchart presented in Fig. 4. Our research work will concentrate mainly in two fronts, i.e., the two final steps of the flowchart displayed in Fig. 4.

In this sense the step "Create Ensemble of Metamodels" of Fig. 4 refers to (i) the creation of accurate metamodels, with limited time and computer resources and the step "Efficient Global Optimization Algorithm" is related to (ii) the management of the whole metamodel based optimization process in an efficient way, as outlined in the closing paragraphs of the last section.

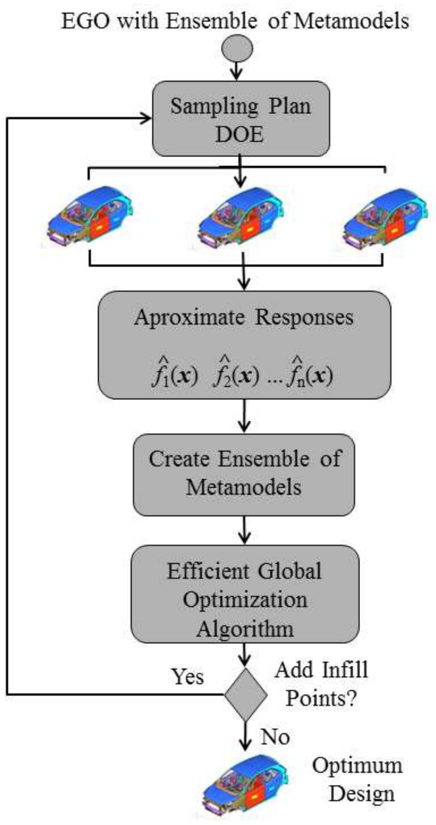

EGO with Ensemble of Metamodels
Sampling Plan DOE
Aproximate Responses
$\hat{f}_1(\boldsymbol{x})$ $\hat{f}_2(\boldsymbol{x})$ ... $\hat{f}_n(\boldsymbol{x})$
Create Ensemble of Metamodels
Efficient Global Optimization Algorithm
Yes
Add Infill Points?
No
Optimum Design

Figure 4: Efficient global optimization (EGO) with ensemble of metamodels. By using ensemble of metamodels the standard EGO (Fig. 3) can be enhanced to accept different approximations (not only kriging) and generate multiple infill points per optimization cycle.

In the next section we will present our specific objectives and main contributions achieved with the work in the research branches discussed here in the field of **Metamodel Based Optimization**, i.e., **Ensemble of Metamodels** and **Efficient Global Optimization**.

1.3 Objectives and Original Contributions of the Thesis

We divided the present thesis work in two main parts, i.e., (i) **Theory and Implementations** and (ii) **Validation and Case Study**. The two main research subjects presented in the previous section, i.e., **Ensemble of Metamodels** and **Efficient Global Optimization** are part of what we called **Theory and Implementations**. These subjects were developed respectively in the **Phase I** and **Phase II** of the work. The **Phase III** refers to the **Validation and Case Study** part of the thesis.

See in Fig. 5 an overall picture of the research subjects discussed and developed in this thesis and their relation with the three main phases that comprise this work.

The "general objective" of this thesis is to compile and report our results and original contributions achieved respectively in the **Phase I**, **II** and **III**. The "specific objectives" and main outcomes with respect to each of these three phases will be described in the sequence.

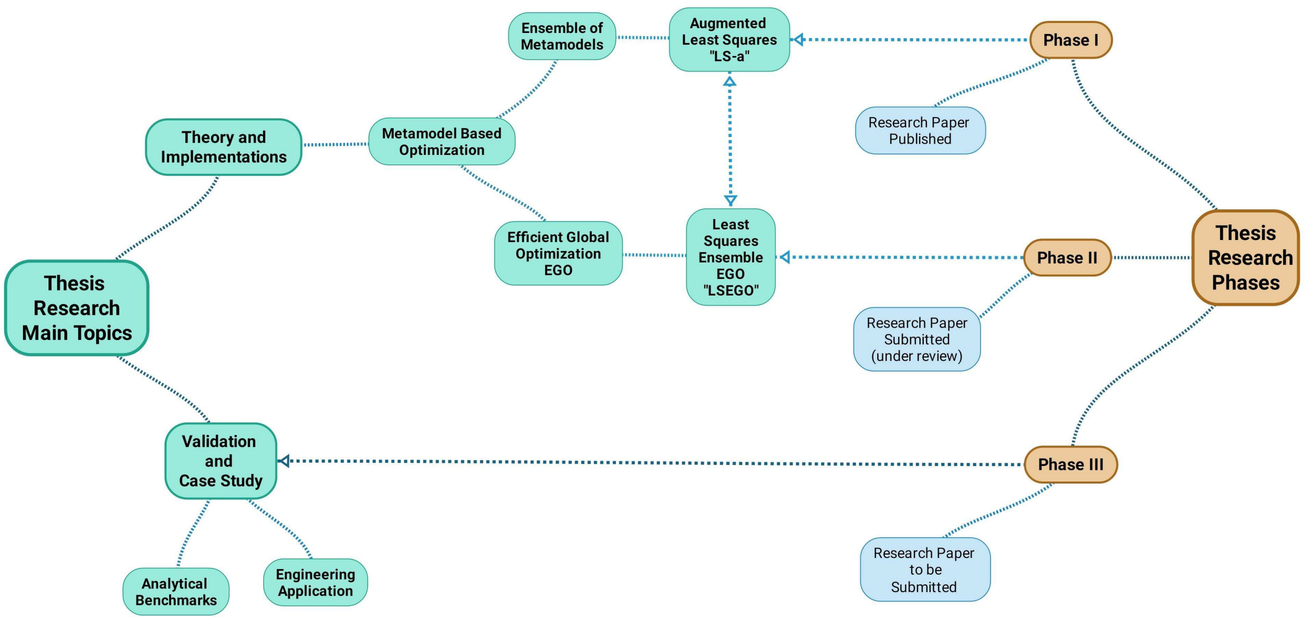

Ensemble of Metamodels
Augmented Least Squares "LS-a"
Phase I
Research Paper Published
Theory and Implementations
Metamodel Based Optimization
Efficient Global Optimization EGO
Least Squares Ensemble EGO "LSEGO"
Phase II
Thesis Research Phases
Thesis Research Main Topics
Research Paper Submitted (under review)
Validation and Case Study
Phase III
Analytical Benchmarks
Engineering Application
Research Paper to be Submitted

Figure 5: Overview of the main research subjects and their relations to the main phases and of the present thesis work.

1.3.1 Phase I

As discussed previously this phase was focused on the research subject **Ensemble of Metamodels**, and the results were published in:

Ferreira W.G., Serpa A.L. (2016) *Ensemble of metamodels: the augmented least squares approach.* Structural and Multidisciplinary Optimization, 53(5), 1019-1046, 2016. DOI: 10.1007/s00158-015-1366-1)

In this first paper we presented an approach to create ensemble of metamodels (or weighted averaged surrogates) based on least squares (LS) approximation. The LS approach is appealing since it is possible to estimate the ensemble weights without using any explicit error metrics as in most of the existent ensemble methods. The proposed LS approach is a variation of the standard LS regression, by augmenting the matrices in such a way that minimizes the effects of multicollinearity inherent to calculation of the ensemble weights.

We tested and compared the “Augmented LS Ensemble” approach (LS-a) with different classical LS variants and also with other existent ensemble methods, by means of analytical and real-world functions from two to forty-four variables. The augmented least squares approach (LS-a) performed with good accuracy and stability for prediction purposes, in the same level of other ensemble methods and has computational cost comparable to the fastest ones.

As an additional feature, the LS based ensemble of metamodels has a prediction variance function that enables the extension to the efficient global optimization. This extension was developed and explored at **Phase II**.

1.3.2 Phase II

This phase was dedicated to the research subject **Efficient Global Optimization**, and the results were submitted for publication. The following paper is under review by the journal editors:

Ferreira and Serpa (SMO-15-0339) *Ensemble of metamodels: Extensions of the least squares approach to efficient global optimization.* Structural and Multidisciplinary Optimization (submitted/under review - ID SMO-15-0339.R1)

In this second paper we presented LSEGO (Least Squares Ensemble EGO), an approach to drive efficient global optimization (EGO), based on the LS (least squares) ensemble of metamodels, that was developed in **Phase I** and reported in our first publication, Ferreira and Serpa (2016).

By means of LS ensemble of metamodels it is possible to estimate the pointwise uncertainty[3] of the prediction by using any kind of metamodel (not only kriging) and provide an estimate for the expected improvement function.

For the analytical problems studied, the proposed LSEGO algorithm has shown to be able to find the global optimum with less number of optimization cycles than the required by the classical single infill point EGO approach.

As more infill points are added per optimization cycle, the faster is the convergence to the global optimum (exploitation) and also the quality improvement of the metamodel in the design space (exploration), specially as the number of variables increases, when the standard single point EGO can be quite slow to reach the optimum.

Therefore the results of **Phase II** has shown that LSEGO can be a feasible option to drive EGO with ensemble of metamodels, and it is not restricted to kriging and to single infill point per optimization cycle. On the other hand, LSEGO was tested and validated only for the unconstrained optimization of a few analytical benchmark functions.

In the **Phase III** of the work we extended the application of LSEGO to constrained optimization problems as well.

1.3.3 Phase III

In the present thesis work we will first review and compile the results of our previous research in the fields of ensemble of metamodels and efficient global optimization (EGO), reported in the papers Ferreira and Serpa (2016) and Ferreira and Serpa (SMO-15-0339), as described above. For easy and quick reference, these two submitted manuscripts are available in the Appendix G.

In addition to these results already reported in **Phase I** and **II**, in the **Phase III** of the work we extended the LSEGO algorithm to handle constrained optimization problems as well, that were not covered in the previous phases.

Some numerical experiments were performed (**Validation**) with analytical benchmark

[3]The *uncertainty* concept treated in the present thesis will be related to only to overall metamodel accuracy, that is mainly driven by different sampling points and different metamodel equations that can be arbitrarily chosen by the user in each problem. It is not objective of the present work to treat uncertainty in the context of Reliability Engineering.

functions and also for an industry scale engineering problem (**Case Study**) achieving competitive results. Our intention is to summarize these last results in a third paper to be submitted for publication near in the future.

In the next section we will outline the chapters structure used to fulfill the objectives and present the main achievements of this research work.

1.3.4 Summary of Implementations

After the end of **Phase I**, **II**, and the **Phase III** of the present thesis, we implement and test algorithms and methods related to:

- Ensemble of Metamodels based on standard least squares approximation and variants (**Phase I**);
- Efficient Global Optimization algorithm based on least squares ensemble of metamodels with extension to constrained optimization (**Phase II** and **Phase II**);

Although the methods developed here are prone to be extended to *multi-objective* optimization, as presented in Forrester et al (2008), in this thesis we only applied and tested the algorithms to *single-objective* unconstrained and constrained optimization problems (i.e., analytical and engineering benchmarks and one industry application).

1.4 Thesis Chapters Outline

In this section it is presented a summary of the main chapters that compose the thesis text. See a brief description of each chapter in the list that follows. Fig. 6 presents the list of the main thesis chapters to the main subjects of the thesis research, as presented and discussed in the previous sections.

- **Chapter 2 - Literature Overview:** The research panorama and key publications available in the literature are presented and discussed.
- **Chapter 3 - Metamodel Based Analysis and Optimization:** The fundamentals and basic formulation of metamodels for analysis and optimization are presented.
- **Chapter 4 - Ensemble of Metamodels Background:** In this chapter it is presented the main formulation with the objective to create ensemble of metamodels found in previous research work available in the literature.

- **Chapter 5 - Ensemble of Metamodels by Least Squares:** This chapter summarizes the main formulations and results achieved in **Phase I** and published in Ferreira and Serpa (2016). This chapter has a close relation to the appendices E and F described later in the sequence.

- **Chapter 6 - Efficient Global Optimization (EGO):** In this chapter it is presented the main definitions and formulation of Efficient Global Optimization algorithms.

- **Chapter 7 - LS Ensemble of Metamodels EGO (LSEGO):** This chapter summarizes the main formulations and results achieved in the **Phase II** of the thesis work. The results were submitted for publication in Ferreira and Serpa (SMO-15-0339).

- **Chapter 8 - Applications to Constrained Optimization Problems:** This chapter refers to the **Phase III** of the thesis work. In this chapter we describe the setup for the numerical experiments performed to solve constrained optimization problems by using LSEGO.

- **Chapter 9 - Results and Discussion:** The results of the numerical experiments described in Chapter 8 are presented and discussed. These are the main results achieved in the **Phase III**.

- **Chapter 10 - Concluding Remarks:** In this chapter we outline all the conclusions of the present thesis and suggest some possible future research directions.

In order to keep the flow of the text as succinct and objective as possible, we moved all the long discussion and derivations to the following appendices:

- **Appendix A - SURROGATES Toolbox:** Presents some details regarding the free SURROGATES Toolbox implemented by F. A. C. Viana and coworkers Viana (2009). All the numerical experiments and developments of this research were based on this toolbox.

- **Appendix B - Boxplots Definition:** Presents the definition of *boxpolt* used to compare and analyze several results of this thesis.

- **Appendix C - Kriging Metamodel:** Kriging metamodel is the basis of EGO implementation (Chapter 6). The summarized formulation and equations are presented in this appendix.

- **Appendix D - Test Functions:** Presents the equations of the test functions referred along the thesis and used in the several numerical experiments performed during the thesis work.

- **Appendix E - Preliminary Numerical Study:** This appendix compiles the first research numerical studies that motivated the development of this thesis and later on culminated in the results and publication achieved in **Phase I**.

- **Appendix F - Multicollinearty and Least Squares:** In this appendix we present the details of the Least Squares methods available in the literature and the methods to handle multicollinearity in regression problems, as referred in Chapter 5. This is an important part of the research of **Phase I** but it is too long to be in the middle of the thesis text.

- **Appendix G - Manuscripts Submitted:** The two already submitted manuscripts, Ferreira and Serpa (2016) and Ferreira and Serpa (SMO-15-0339) are completely available in this appendix for easy and quick reference.

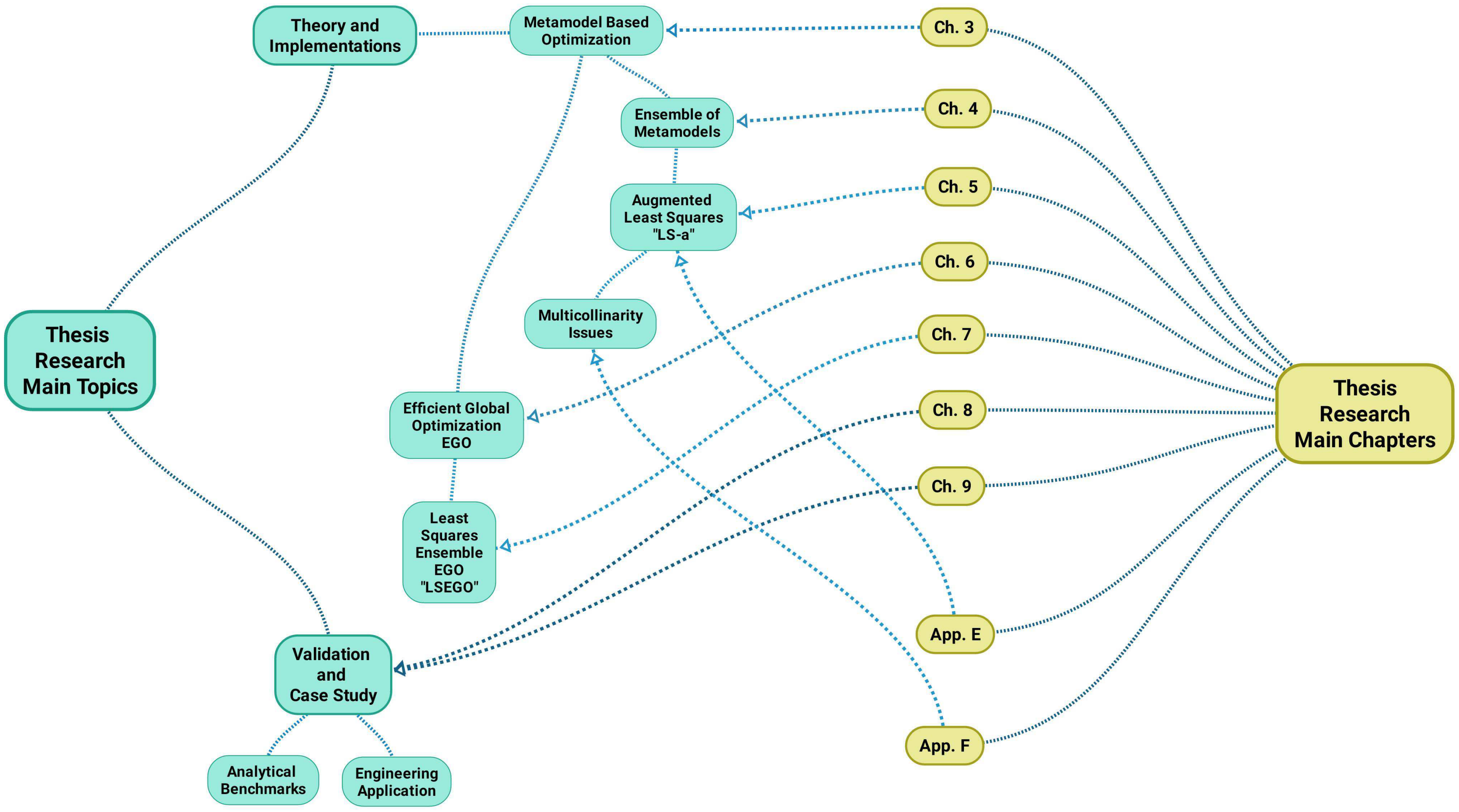

Theory and Implementations
Metamodel Based Optimization
Ch. 3
Ensemble of Metamodels
Ch. 4
Augmented Least Squares "LS-a"
Ch. 5
Ch. 6
Multicollinarity Issues
Ch. 7
Thesis Research Main Topics
Efficient Global Optimization EGO
Ch. 8
Thesis Research Main Chapters
Ch. 9
Least Squares Ensemble EGO "LSEGO"
App. E
Validation and Case Study
App. F
Analytical Benchmarks
Engineering Application

Figure 6: Overview of the main chapters and their relations to the main subjects of the thesis research. For clarity, the chapters related to all subjects were omitted in the picture, i.e., Introduction, Literature Overview, Concluding Remarks and remaining appendices.

2 Literature Overview

"If I have seen further it is only by standing on the shoulders of giants..."
Isaac Newton, 1642-1727.

2.1 Some Historical Remarks

The development of methods to describe the functional representation of scattered data remounts to the beginning of the natural sciences. The mathematicians K. F. Gauss (1777-1855) and P. S. Laplace (1749-1827) proposed different methods for estimating functional dependencies from results of measurements of astronomy and physics. Gauss proposed the Least Squares Method (LSM) and Laplace proposed the Least Modulo Method (LMM) and since that time a question has raised to define which method was the best.

Until the beginning of the twentieth century preference was given to LSM. Later, in the mid of twentieth century, it was observed that the accuracy of any estimator method is strongly dependent on the noise pattern, its distribution and relationship with the measured data. These observations favored the preference for LMM in some applications.

Other methods like Maximum Likelihood Method (MLM) has been proposed (R. Fisher, 1920) in order to cope with noise and the concept of *robust estimation* was introduced by P. Huber in the 1960s.

Until nowadays it is not clear what is the best strategy for estimating functions in real-life situations specially when the form and distribution of noise are not known. For further details on these early developments related to approximation methods see Vapnik (2000) and the references therein.

Based on literature survey published by Barthelemy and Haftka (1993), the use of approximation methods in conjunction with nonlinear programing to solve large structural design problems has been proposed in earlier studies of Schmit, Arora and co-workers in the middle of 1970s (e.g., Schmit and Farshi (1974), Schmit and Miura (1976) and Arora (1976)).

As discussed in this review by Barthelemy and Haftka (1993), at the beginning of 1990s there were not enough data in the open literature to establish comprehensive comparison of the effectiveness of various approximation concepts applied to structural optimization. The selection of which was the best method should be based on several attempts of different techniques suited for the problem at hand and in the most cases the better solution was a combination of different approaches.

In this sense, Barthelemy and Haftka (1993) concluded that "newer and emergent methods" like construction of response surfaces (e.g. Sacks et al (1989a) and Sacks et al (1989b)) and neural networks (e.g., Hajela and Berke (1990) and Carpenter and Barthelemy (1992)) should become cost effective even for large-scale problems with the advent of massively parallel computers. These global approximation methods should also provide unique opportunities to build inexpensive approximations to expensive analytical models[4].

Most of the early literature related to metamodeling are traditionally written in the statistics and operations research context, as example Box (1954), Box et al (1978), Kleijnen (1985) and more recently Myers and Montgomery (2002) and Santner et al (2003).

Simpson et al (2001a) present a detailed survey of metamodels for computer-based engineering design. They reviewed the common statistical techniques used to build approximations of computer analysis codes in engineering design at that time. Also they were worried about the "dangers of applying traditional statistical techniques to approximate *deterministic* computer analysis codes".

Based on our research of the open literature, Simpson et al (2001a) is one of the first works that tried to define a clear and systematic methodology for the application of metamodeling techniques to *deterministic* computer-based engineering design and to differentiate it from traditional statistics methods applied mainly to real-world experiments, that are subjected to random error in parameters and output, i.e., *stochastic* problems.

Simpson et al (2001a) reviewed the current available experimental design techniques and its measures of merit, the prevalent approximation strategies named: response surfaces (i.e., polynomial regression), neural networks, inductive learning, kriging, and alternative new choices for fitting such as multivariate regression splines and radial basis functions. Also, they discussed the Taguchi approach combined to response surfaces in robust engineering design. At the end, they summarized some guidelines and recommendation on using metamodels with deterministic computer codes.

Simpson et al (2001a) advocated that comprehensive comparisons among different ap-

[4]After the work of Sacks et al (1989a) this approximation process become known as DACE, design and analysis of computer experiments. Although the use of DACE is becoming rare nowadays, this term still can be found in many references.

proaches should be performed in order to better understand its advantages and disadvantages. The difficulties to handle problems with large-scale (i.e., more than 10 variables) were highlighted and they suggested further investigations of problem partitioning or domain decomposition methods to deal with large-scale problems. Finally, they suggested research on metamodeling techniques applied to multi-objective and multidisciplinary engineering design analysis and optimization.

In Queipo et al (2005) it is presented a comprehensive discussion of the fundamental aspects of surrogate-based (or metamodel-based) analysis and optimization, emphasizing main theoretical concepts, methods, techniques and practical implications. Also, it is presented a case study of the multi-objective optimization of an aerospace component by using polynomial regression. Among research issues related to surrogate-based analysis and optimization they listed: improvements on sampling techniques, alternative modeling methods to lead the development of more robust approximation schemes (e.g., support vector regression), the use of multiple surrogates simultaneously in prediction and optimization, more effective methods for model validation (e.g., bootstraping) and efficient strategies for sensitivity analysis and design screening. At the end, they highlighted that surrogate-based analysis and optimization is an active area of research that has "the potential of playing a vital role in the successfull full-scale development or modern aerospace systems while effectively considering the competing needs of improving performance, reducing costs, and enhancing safety".

Fang et al (2006) published the book: *Design and Modeling for Computer Experiments*. The differential aspect of this book is that it is devoted to computer experiments, applied mainly in approximation and optimization of computer-based simulated *engineering problems* and not to statistical research. In fact, most of the examples in their book are collected from engineering applications, from scientists and practitioners in industry.

In the same way, Forrester et al (2008) published another important book in the field: *Engineering Design via Surrogate Modeling: A Practical Guide*, which confirms the increasing interest and the technological importance achieved by this area during the previous years for the engineering community.

Simpson et al (2008) presented an extensive historical survey and they recall that a central question in the metamodeling for design and optimization is when multiple models are to be considered. So the issue is on what approach to use: selecting or combining models. However, they remarked that, up to that point, the advantages of combination over selection have never been clarified. Within the controversy is that the most accurate surrogate did not always lead to the best design, thus using multiple surrogates (or ensemble of surrogates) can improve the robustness of the optimization at a minimal computational cost. On the other hand, it is mentioned that after some research in the area, the potential gains of weighted average ensemble of surrogates should diminish substantially in high dimension problems, possibly making the gains very difficult in practice.

Besides all the controversy, Simpson et al (2008) remark that since the use of multiple surrogates for optimization is affordable when compared to the actual simulations, then the use of a set of surrogates (and possibly a weighted average surrogate) is very appealing in design optimization research and practical applications.

Forrester and Keane (2009) published another extensive review in surrogate based optimization area. They discussed several techniques available for constructing surrogates and also some enhancing possibilities such as exploiting gradient information and also multi-fidelity analysis techniques. In addition, they discussed in detail some infill criteria available to improve the optimization process and to balance exploration and exploitation capabilities provided by the surrogate approach. In this sense, it is worth noting the Efficient Global Optimization (EGO), based on probability of improvement and expected improvement functions, after the pioneering work of Schonlau (1997) and Jones et al (1998). Finally, they remarked that the metamodel based optimization must always include some form of iterative search and repetitive infill process to ensure the accuracy in the areas of interest in the design space.

After developments in the last four decades, the use of metamodeling methods is nowadays a common place in both research and practice in engineering design, analysis and optimization. For a broader and historical perspective on this subject, refer for instance: Schmit and Farshi (1974), Arora (1976), Kleijnen (1985), Sacks et al (1989a), Barthelemy and Haftka (1993), Madu and Kuei (1994), Roux et al (1998), Wang and Shan (2007), Simpson et al (2008), Forrester and Keane (2009), Viana et al (2010), Han and Zhang (2012) and Ramu and Prabhu (2013). In addition, a collection of engineering research and applications has been recently published in Koziel and Leifesson (2013).

As discussed in the previous chapter (Section 1.2), this research is concentrated on **Metamodel Based Optimization** and, for our purposes here, this branch of research was divided in two other branches, i.e., **Ensemble of Metamodels** and **Efficient Global Optimization**. The literature regarding theses subjects is vast, but we can select some key references (about thirty papers and books) that play an important role to the development of the present research and thesis. With no intention to create an ultimate and definitive list, in Fig. 7 it is presented an overview of the main references and their relations to the main subjects studied in the present thesis.

Although metamodeling applied to simulation-based engineering design and optimization is a recognized mature discipline, it is yet an intensive research field with some open issues and controversy to be resolved. It is a consensus (e.g., Viana et al (2010) and Yang et al (2013b)) that there are some areas demanding further studies and enhancements, for instance:

- Development of more effective sampling methods;
- Feasibility and efficient treatment of large-scale problems;
- Alternative modeling methods to lead the development of more robust approximation schemes (e.g support vector regression);
- Efficient handling of several responses and constraints;
- More effective methods for model validation;
- Improvements and new strategies for sensitivity analysis, results visualization and design screening;
- Efficient methods to capture and deal with uncertainty, reliability and robustness.
- Efficient methods for combining EGO algorithms and iterative search (repetitive infill process) to ensure the accuracy in the areas of interest in the design space.

In the next sections we will provide some more details and definitions regarding **Metamodel Based Optimization**, **Ensemble of Metamodels**, **Efficient Global Optimization** with the specific literature that provides the background and motivations for the research presented in this thesis.

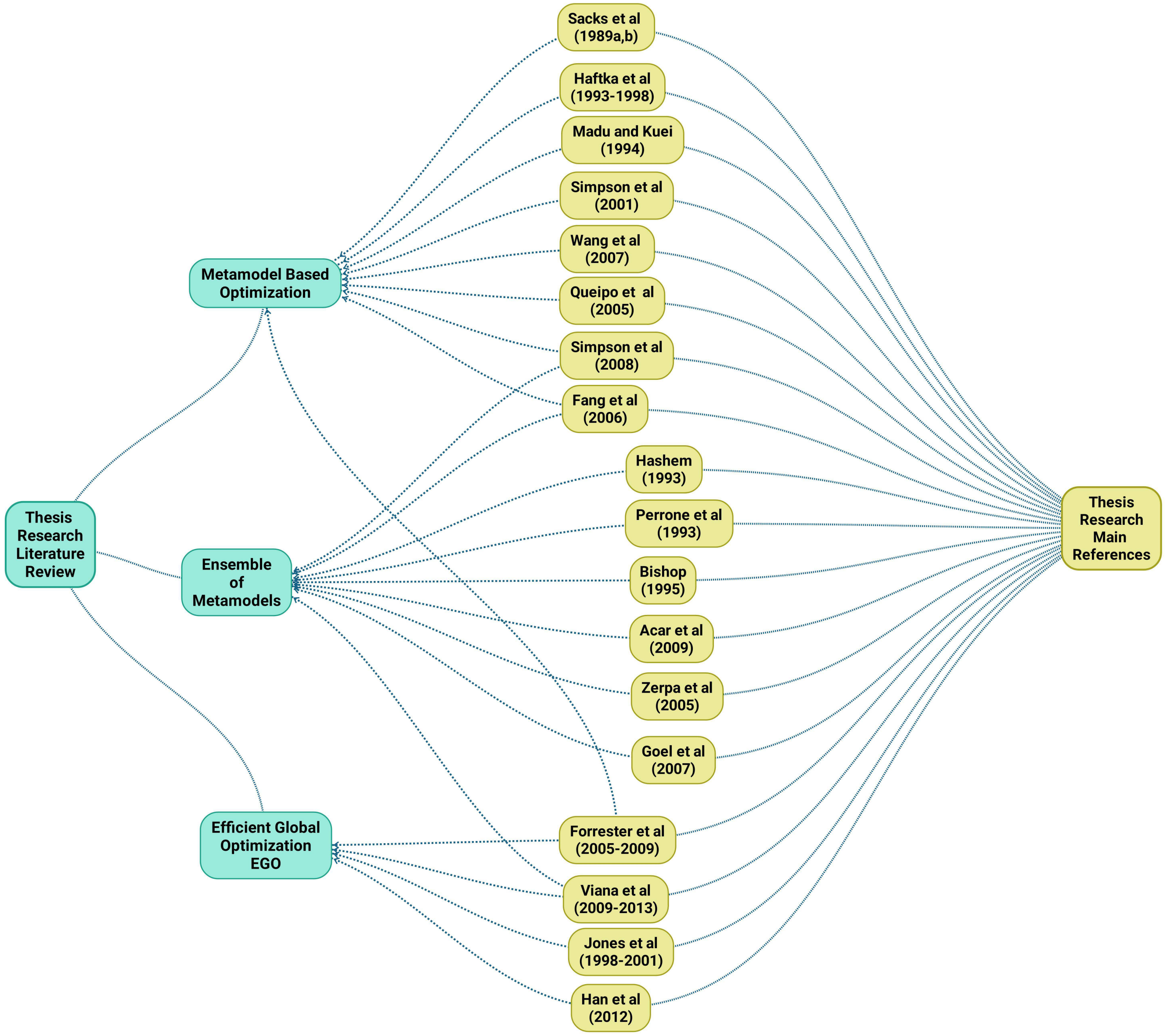


Figure 7: Overview of the main literature references and their relations to the main research subjects of the thesis.

2.2 Specific Literature for the Thesis Scope

In the Section 1.2, the research scope of this thesis work was established and the main concepts and the basic literature references were outlined. In order to develop the theoretical concepts and implementations regarding **Metamodel Based Optimization**, **Ensemble of Metamodels**, **Efficient Global Optimization**, each branch must be divided in new sub-branches.

A more detailed view of the research subjects studied in the present thesis is displayed in Fig. 8. As can be observed in the **Theory and Implementations** part, the two main paths in direction of **Ensemble of Metamodels** and **Efficient Global Optimization** are highlighted.

In the next sections the state of the art regarding these two main paths followed in the present thesis will be detailed. The specific literature and main authors will be presented and the relations to the implementations developed during this research, as outlined in Section 1.3, i.e., the Augmented Least Squares Ensemble (LS-a) and the Least Squares Ensemble EGO (LSEGO).

The definitions, formulations and equations regarding each subtopic displayed in Fig. 8 will be presented in the respective chapters and appendices, as described in Section 1.4.

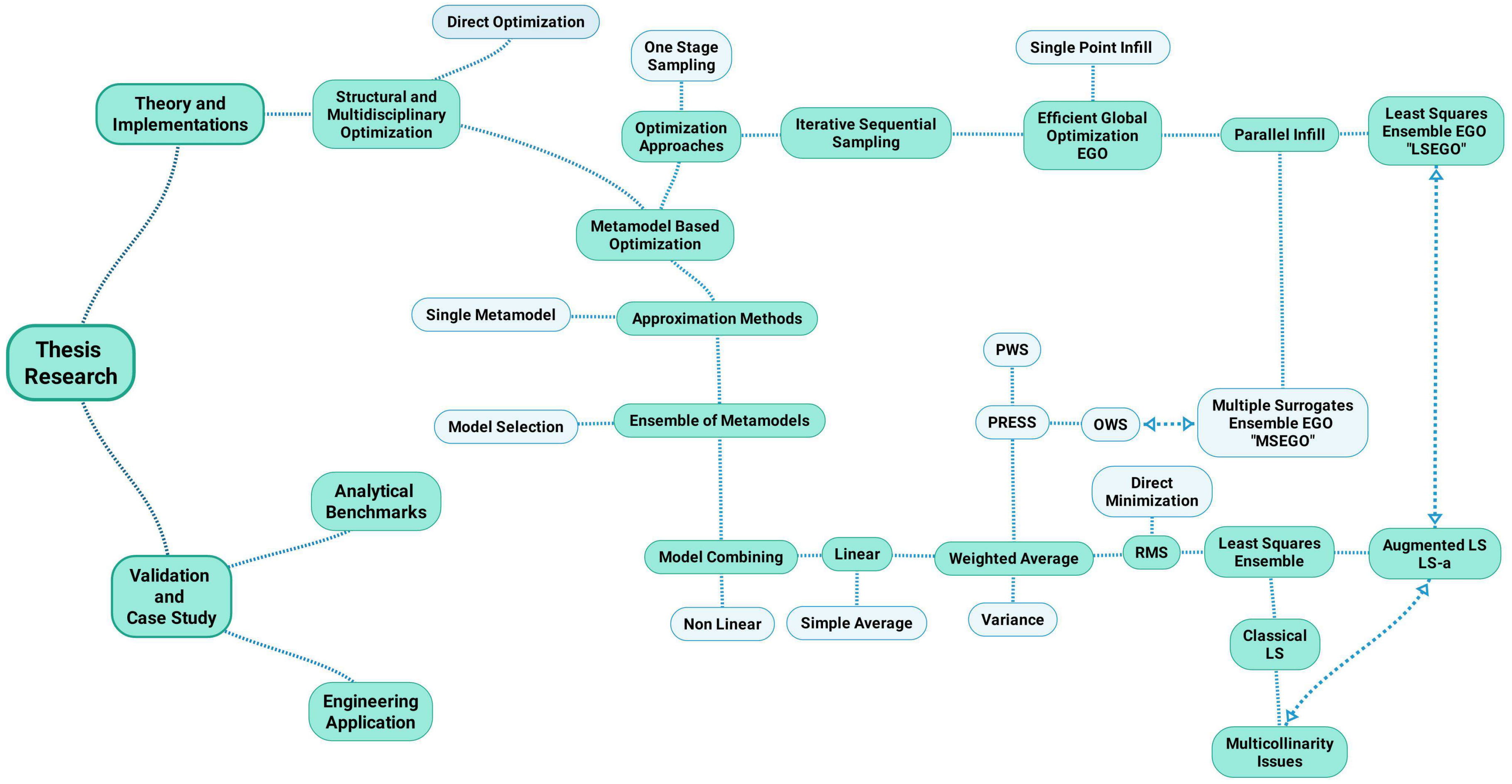


Figure 8: Breakdown of the main research subjects of the thesis. Acronyms: RMS (root mean squares), PRESS (predicted sum of squares), PWS (PRESS weighted surrogate), OWS (optimal weighted surrogate), LS (least squares).

2.2.1 Metamodel Based Optimization

Let us recover the brief discussion of Section 1.2. For convenience Fig. 9 will reproduced here in order to keep the present section self contained.

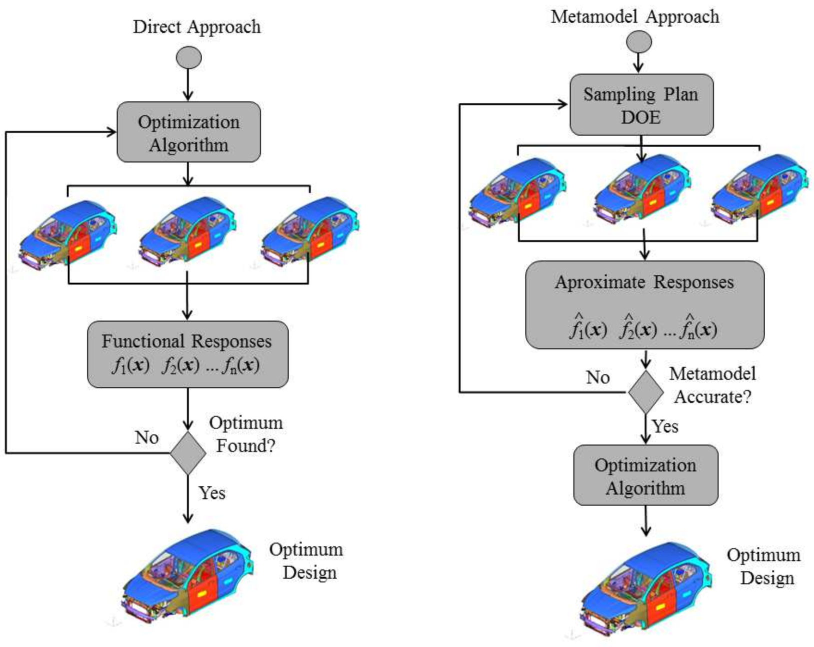


Figure 9: *Direct Optimization* (left) versus *Metamodel Based Optimization* (right). In the *direct* approach, at each iteration or cycle, the optimization algorithm submit the simulation models to run in order to evaluate the performance responses $f_i(\mathbf{x})$. This process can be repeated for several iterations, with hundreds or thousands function evaluations. In the *metamodel based* approach, a design of experiments (DOE) is created and the simulation models are submitted to evaluate the performance responses $f_i(\mathbf{x})$. This process is repeated until reasonable approximations $\hat{f}_i(\mathbf{x})$ for the responses are achieved. At the end, the optimization algorithm uses the metamodels $\hat{f}_i(\mathbf{x})$ to find the optimum design. In practical terms, the direct approach is prohibitive (or even impossible) to be applied in MDO problems as in Fig.1. On the other hand, the metamodel approach is feasible and efficient in such complex MDO scenarios.

As we stated before, in simple terms, metamodels or surrogate models are nothing but some kind of "easy-to-calculate" fitting functions (e.g., interpolated or regression polynomials), constructed with a set of data samples available. In this sense, the sampling data (i.e., different simulation models in a DOE) can be evaluated in parallel and the overall optimization process should be faster and it should require less function evaluations than the direct approach to find the global optimum or even a set of improved design solutions.

The surrogate based optimization is in general an iterative (cyclic) process. At each cycle, instances of simulation models (sampling points) with different parameters are evaluated. The surrogate models are fit based on these data and the resulting approximate functions are used in the search of optimum points (exploitation), analysis of the response behavior, sensitivity and trends in the design space (exploration). Once optimum design candidates are found, they are evaluated with the true simulation models and, if necessary, the new points are included in the sampling space in order to improve the approximation and to restart the iterative process.

In summary, surrogate based (or response surface based, or metamodel based) optimization refers to the process of using fast running metamodels as surrogates of original complex and long time running computer simulation models to approximate the objectives and constraint functions in a standard optimization algorithm. This methodology has shown to be effective in both multidisciplinary and multiobjective optimization problems and it has been widely applied in research and industry. Refer to the reviews by Queipo et al (2005), Wang and Shan (2007), Simpson et al (2008) and Forrester and Keane (2009) for the mathematical background and detailed discussion on this subject.

2.2.2 Efficient Global Optimization

Recall in Fig. 10 the basic flowchart for the EGO approach, also already introduced in Section 1.2.

The EGO concept emerged after the work of Jones et al (1998), that was based mainly on previous research on Bayesian Global Optimization (Schonlau (1997) and Mockus (1994)). Traditionally the available EGO algorithms are based on kriging approximation and a single infill point is generated per optimization cycle.

A question that arises is that the selection of only one infill point per optimization cycle can be quite slow to achieve the convergence to the optimum. If parallel computation is an available resource, then multiple infill points should be defined and therefore less cycles might be required for convergence. This aspect can be crucial specially if the computer models take several hours to run (as we pointed out in Section 1.1) and a single point EGO approach becomes prohibitive, specially in multidisciplinary optimization scenarios.

In practical terms, when parallel resources are easily available, it can be worthwhile to run more simulations per cycle in order to reach an optimum in a reasonable lead time, even if the total number of simulations is higher at the end of the optimization process.

The aspect of single versus multiple infill points per cycle is known and discussed since the origins of EGO-type algorithms in the later 1990s. Schonlau (1997) proposed extensions to the standard EGO algorithm to deliver "m points per stage", but he pointed out numerical difficulties in the evaluation of the derived expressions accurately at a reasonable computer cost

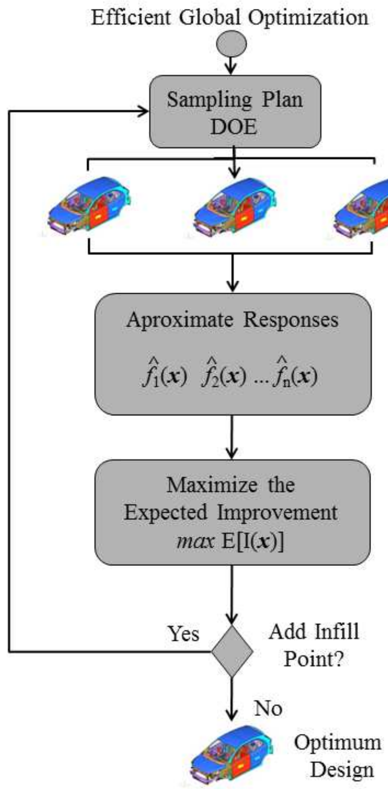

Efficient Global Optimization
Sampling Plan DOE
Aproximate Responses
$\hat{f}_1(x)$ $\hat{f}_2(x)$ ... $\hat{f}_n(x)$
Maximize the Expected Improvement *max* E[I(*x*)]
Yes
Add Infill Point?
No
Optimum Design

Figure 10: Efficient global optimization (EGO) flowchart. EGO is a sequential sampling metamodel based optimization approach. Metamodels $\hat{f}_i(\mathbf{x})$ are created and the expected improvement functions $E[I(\mathbf{x})]$ are maximized to generate infill points for the next optimization cycle. The optimum is found if the level of expected improvement is negligible or the maximum allowed number of cycles iterations is reached.

and number of function evaluations.

Besides this apparent disadvantage in terms of function evaluations, it can be observed an increasing research interest on EGO approaches with multiple infill points per cycle (and other metamodel based parallelization strategies as well) published in the last ten years. This is because parallel computation is nowadays a relatively easy resource and the potential penalty of parallel approaches in terms of function evaluations should be neglected in favor of quickly delivering optimization results.

Although it can be considered a relatively new research field surrogate based global optimization is gaining popularity as pointed out in Haftka et al (2016). In this recent and broad survey they examined and discussed the publication focused on parallel surrogate-assisted global optimization with expensive models. According to the authors this area is not mature yet and it is not possible to conclude with respect to the comparative efficiency of different approaches or algorithms without further research.

As discussed by Haftka et al (2016) different classes of algorithms or strategies can be defined with the objective of balancing exploitation and exploration of the design space during the optimization. In simple terms exploitation refers to deep diving (or zooming) in the candidate areas of feasible optimum points in order to improve the objective function and the constraints to deliver better optimization results. On the other hand, exploration means adding infill points in different areas of the design space in order to reduce the uncertainty and to improve the metamodel prediction capability.

The different classes of strategies involve the ones based on nature inspired algorithms like genetic, evolutionary, particle swarm, etc., that are naturally parallelized (the populations can be divided in different regions, or processors) and are commonly applied in global optimization. In addition surrogate based strategies (like EGO) can be used together with the parallelization to improve the exploitation and exploration features of the algorithms.

In this sense, specifically in the branch of metamodel-based multiple points per cycle algorithms (like EGO or other approaches with similar objectives), refer for instance to Sóbester et al (2004), Henkenjohann and Kukert (2007), Ponweiser et al (2008), Ginsbourger et al (2010), Viana and Haftka (2010), Janusevskis et al (2012), Viana et al (2013), Desautels et al (2012), Chaudhuri and Haftka (2012), Rehman et al (2014), Mehari et al (2015), Encisoa and Branke (2015) and other interesting and relevant works referenced and discussed in Haftka et al (2016).

2.2.3 Ensemble of Metamodels and EGO

The effectiveness of *selecting* and/or *combining* different metamodels in the optimization process has been investigated and discussed in the last years. See, for instance Zerpa et al (2005), Goel et al (2007), Sanchez et al (2008), Viana et al (2009), Acar and Rais-Rohani (2009), Acar (2010) and Viana (2011).

In most of these studies, it is suggested that *ensemble of metamodels*, or *weighted averaged surrogate* (WAS) models, are able to provide better accuracy than individual metamodels working alone, and they can improve the overall robustness of the metamodel based optimization process. Besides that, in Viana et al (2009) and Viana (2011) it is discussed that the potential gains of using multiple surrogate models are not guaranteed and should be limited.

In fact, as pointed out in Viana (2011), "even after years of intensive research, surrogate modeling still involves a struggle to achieve maximum accuracy within limited resources". In addition, as discussed by Yang et al (2013b), among the challenges in modeling and optimization in the next years, (i) the best way to construct good surrogate models and (ii) the choice of modeling and optimization algorithms for a given problem are still open questions for research.

As mentioned in Section 1.3, we proposed in the **Phase I** of this research the use of the concept of least squares (LS) regression as a way to find the optimal weights in ensemble of metamodels (or weighted average surrogates, WAS). These implementations and results were published in Ferreira and Serpa (2016).

In most of the currently available WAS methods, for instance Goel et al (2007), Viana et al (2009) and Acar and Rais-Rohani (2009), it is necessary to use specific error metrics like PRESS (prediction sum of squares) to drive the process of finding the best weights in the ensemble of metamodels. One of the drawbacks is that PRESS often has a high computational cost to be evaluated.

Therefore, the LS approach has shown to be appealing in terms of lower computational cost and simple formulation. In addition, LS methods are well known and established in statistics field and have a series of variants developed to handle, for example, multicollinearity, an inherent drawback that limits the accuracy of ensemble of metamodels.

Specifically for handling multicollinearity, we proposed in Ferreira and Serpa (2016) the *augmented least squares ensemble of metamodels* (LS-a), that is a variation of the standard least squares regression, by augmenting the matrix system in such a way that minimizes the effects of linear dependency among the models.

In a second front of application, we pointed out that LS ensemble methods can be used within the context of efficient global optimization. The ensemble of metamodels based on LS approach inherits the variance estimator, which can be used in the definition of the *expected improvement function*, that is the main driving force behind EGO methods.

In the **Phase II** of the present research, we extended our first results by proposing an approach that is able to provide multiple infill points per cycle in a EGO-type algorithm by using least squares ensemble of metamodels.

In Ferreira and Serpa (SMO-15-0339), see Appendix G, we presented the LSEGO algorithm, that stands for "least squares ensemble efficient global optimization". In the numerical experiments performed, LSEGO has shown to be a feasible alternative to drive EGO algorithms with ensemble of metamodels, and in addition it is not restricted to kriging nor a single infill point per optimization cycle. LSEGO was applied in the optimization of analytical benchmark functions (up to six variables) and including some examples of constrained optimization as well. In addition LSEGO produced competitive results as compared to MSEGO for the optimization

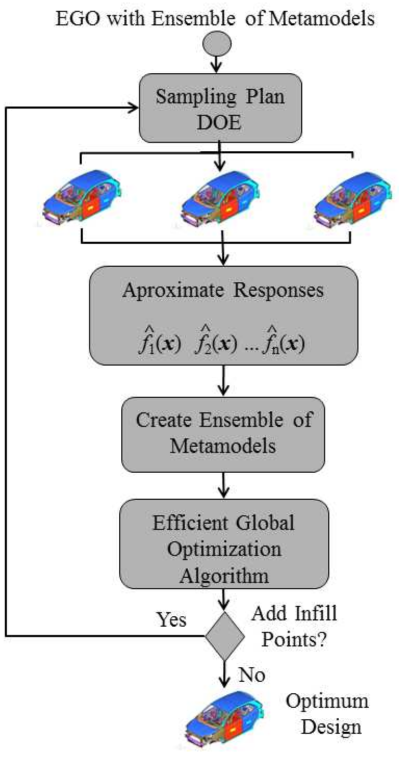

EGO with Ensemble of Metamodels
Sampling Plan DOE
Aproximate Responses
$\hat{f}_1(\boldsymbol{x})$ $\hat{f}_2(\boldsymbol{x})$ ... $\hat{f}_n(\boldsymbol{x})$
Create Ensemble of Metamodels
Efficient Global Optimization Algorithm
Yes
Add Infill Points?
No
Optimum Design

Figure 11: Efficient global optimization (EGO) with ensemble of metamodels. By using ensemble of metamodels the standard EGO (Fig. 10) can be enhanced to accept different approximations (not only kriging) and generate multiple infill points per optimization cycle, as in LSEGO algorithm reported in Ferreira and Serpa (SMO-15-0339), see Appendix G.

of two benchmark functions. Recall in Fig. 11 a flowchart that summarizes the EGO process with ensembles, as in LSEGO, previously mentioned in Section 1.3 as well.

Viana et al (2013) proposed the MSEGO (multiple surrogates EGO). They devised a way export the uncertainty estimate for one kriging model to the other non-kriging models in a multiple surrogates set. With different uncertainty estimates, they generate different instances of expected improvement functions to be maximized and to provide multiple parallel infill points in each EGO cycle. We suggested an alternative scheme to generate multiple infill points per cycle in the EGO algorithm by using a LS ensemble, in order to take advantage of multiple surrogates, as applied in Viana et al (2013).

In the present thesis we review and compile the results of our previous research in the field of LS ensembles and EGO algorithms. In addition, in the **Phase III** of the present work, we extended the LSEGO algorithm to handle constrained optimization problems. In order to test the effectiveness of LSEGO approach in constrained optimization, some numerical experiments were performed with analytical test functions. In addition, a case study with one industry scale engineering problem is also presented.

3 Metamodel Based Analysis and Optimization

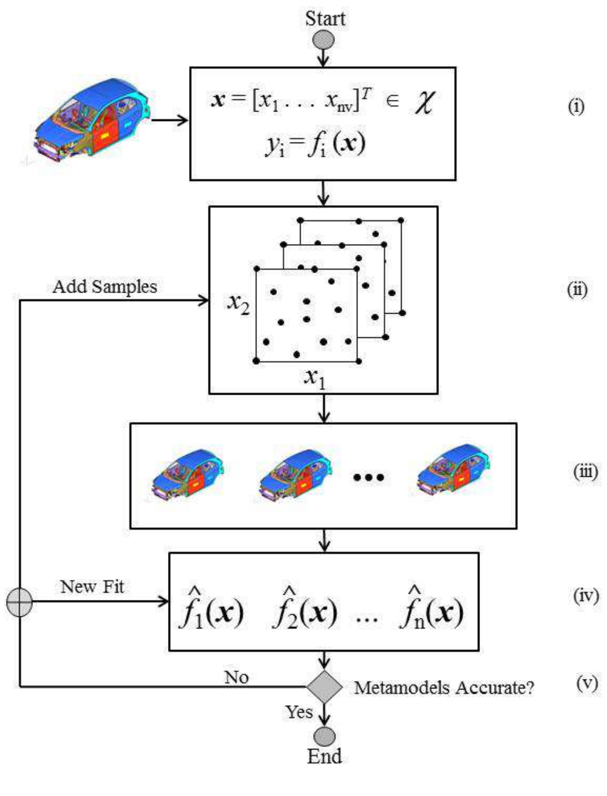
"It's better to solve the right problem approximately

than to the wrong problem exactly."

J. W. Tukey, 1915-2000.

Most physical phenomena can be described by means of mathematical models, such as $y = f(\mathbf{x})$, where $\mathbf{x}$ represent a vector of input variables, defined in $\Re^n$, and y is an output scalar variable that represents the response $f(\mathbf{x})$ being modeled.

Since $f(\mathbf{x})$ is costly and/or time consuming to evaluate, as in computer models of real world applications (e.g., FE and CFD simulation models), the idea is to find a proper approximation $\hat{y}(\mathbf{x}) \approx y(\mathbf{x})$, also known as metamodel, surrogate or response surface, that is at same time accurate, cheap and fast to evaluate.



Start
$\boldsymbol{x} = [x_1 \ldots x_{nv}]^T \in \chi$
$y_i = f_i(\boldsymbol{x})$
(i)
Add Samples
x_2
x_1
(ii)
(iii)
New Fit
$\hat{f}_1(\boldsymbol{x})$ $\hat{f}_2(\boldsymbol{x})$ ... $\hat{f}_n(\boldsymbol{x})$
(iv)
No
Metamodels Accurate?
(v)
Yes
End

Figure 12: The iterative process of creating metamodels: (i) Definition of Design Space, (ii) Experimental Design, (iii) Models Evaluation, (iv) Metamodels Creation and (v) Metamodels Validation. At the end of the process, the true response functions (original models) $y_i = f_i(\mathbf{x})$ can be replaced by validated approximate functions (metamodels) $\hat{y}_i = \hat{f}_i(\mathbf{x})$, that are accurate and fast to evaluate in sensitivity analysis, design space exploration and optimization studies.

The process of constructing approximate models (i.e., metamodeling) is iterative by definition. The five main steps that cover the metamodeling process (see Fig. 12) should be briefly described as follows:

(i) ***Definition of Design Space*:** based on the original simulation models available, select the *inputs*, i.e., the n_v design variables $\mathbf{x} = [x_1 \ \cdots \ x_{n_v}]^T$ and respective bounds of design the space, i.e., $\mathbf{x} \in \chi$. In addition, select the *outputs*, i.e., the responses $y_i = f_i(\mathbf{x})$ to be monitored. This step can be driven by prior experience and knowledge regarding the problem, or by a previous sensitivity analysis to prioritize the most significant n_v variables, respective bounds and associated functional responses to control or optimize. Although it is an important aspect for the proper and reasonable definition of variables and the design space, discussion of sensitivity analysis methods is out of the scope of this thesis. The interested reader should refer to Morris (1991), Saltelli et al (2004), Ioss and Lemaitre (2015) and the references therein;

(ii) ***Experimental Design*:** perform a design of experiments (DOE), e.g., Factorial Design, Uniform Design, Latin Hypercube, etc., and generate a set of N sampling points (or "training" points) to evaluate the true functions $y_i(\mathbf{x})$;

(iii) ***Models Evaluation*:** run the true models and evaluate $y_i(\mathbf{x})$ at the *training* points defined in step (ii);

(iv) ***Metamodels Creation*:** create the metamodels $\hat{y}_i(\mathbf{x})$ by using any available procedure, e.g.: PRS, Polynomial Response Surface; RBF, Radial Basis Functions; KRG, Kriging; NN, Neural Networks; SVR, Support Vector Regression, etc.;

(v) ***Metamodels Validation*:** define a suited error metric and test the accuracy of $\hat{y}_i\left(\mathbf{x}\right)$ on the *validation* points. If the accuracy is not acceptable, iterate by adding more sampling points, step (ii); or by improving the approximation method (new fit or fine tuning), step (iv).

In this way, by means of validated metamodels $\hat{y}_i(\mathbf{x})$, many engineering activities can be performed in a faster way, such as: preliminary studies and graphical visualization; prediction and optimization; design sensitivity analysis; probabilistic, robust and reliability based design.

The metamodeling process for design and optimization has been developed and matured during the last three decades. All the five steps outlined above and depicted in Fig. 12 are well established in engineering design, and the details can be found in the books by Fang et al (2006) and Forrester et al (2008).

3.1 Metamodel Creation Methods

In a general form, most metamodels can be created by using a linear combination of a set of specific basis functions $\mathbf{B} = \{B_1(\mathbf{x}),\ B_2(\mathbf{x}), \cdots, B_L(\mathbf{x})\}$, as described by Fang et al (2006). Therefore, the metamodel $\hat{y}(\mathbf{x})$ can be written in the following form:

$$\hat{y}(\mathbf{x}) = \sum_{i=1}^{L} \beta_i B_i(\mathbf{x}), \tag{1}$$

where β_i are unknown coefficients to be calculated.

There are different methods for selecting the basis set and calculating the coefficients, which lead to different properties, such as: conceptual simplicity, transparency, accuracy, robustness, efficiency and computational cost. A detailed explanation about these methods should be found in the books by Fang et al (2006) and Forrester et al (2008).

In engineering analysis and optimization the most common methods are:

Polynomial Response Surface (PRS): it is the oldest and one of the most popular technique in metamodeling, also known as Response Surface Methodology after Box (1954). Further details are available in the works of Box and Draper (1987) and Myers and Montgomery (2002).

Kriging (KRG): named after Krige (1951) in geostatistics work, also known as spatial correlation modeling, has become popular in design and analysis of computer experiments after Sacks et al (1989b). A recent comprehensive review on kriging metamodeling can be found in Kleijnen (2009). Since it is the basis of EGO methods (Chapter 6), the main equations are presented for reference in Appendix C.

Radial Basis Functions (RBF): initially developed by Hardy (1971), also in geology field, it is one of the most often applied approaches in modern multivariate approximation theory, when the task is to approximate (interpolate) scattered data in several dimensions. Further details can be found in Buhmann (2003), Wendland (2005) and Fasshauer (2007).

Neural Networks (NN and RBNN): according to Fang et al (2006) the term *neural network* has evolved to encompass a large class of models and "learning" methods for parameter estimation. The architecture of a neural network, based on the concept of *neuron* *model*, describes how a network transforms its input to an output, as a mapping based on weights that can be viewed as a non-parametric regression. The NN literature is large and a broad description can be found in Bishop (1995). There is a close relation between RBF and NN, see for instance a discussion in Buhmann (2003). A particular case of NN-RBF is RBNN, radial basis neural networks, in which the resulting metamodel is a regression instead of an interpolation.

Support Vector Regression (SVR): it is a special case of Support Vector Machines, that is a generalization of the *generalized portrait algorithm*, developed by Vapnik and Lerner (1963). A detailed explanation of SVR is available in Gunn (1997) and Smola and Schölkopf (1998).

Nowadays, it is well known among the researchers and practitioners that there is no "universal" approximation method for metamodeling purposes. The best strategy is dependent on: the degree of nonlinearity of the problem at hand; number of design variables; on the sampling technique; the functional form of surrogate model (shape and tuning parameters) and the error measure adopted.

Interesting discussion can be found in the surveys on this branch of research, also referred as "Design and Analysis of Computer Experiments", "Metamodel Based Analysis and Design" or yet "Surrogate-based Analysis and Optimization": Queipo et al (2005), Wang and Shan (2007), Simpson et al (2008), Forrester and Keane (2009), Viana et al (2010), Han and Zhang (2012) and Ramu and Prabhu (2013).

3.2 Metrics for Metamodel Validation

Regarding the validation step in the metamodeling process outlined previously (see Fig. 12), several metrics can be defined to evaluate the prediction error $e(\mathbf{x}) = y(\mathbf{x}) - \hat{y}(\mathbf{x})$, for example: maximum absolute, average, correlation coefficient, etc. One common error measure adopted is the *root mean squared error*, given by

$$RMSE = \sqrt{\frac{1}{N_{test}} \sum_{i=1}^{N_{test}} e_i^2(\mathbf{x})} \ , \tag{2}$$

where N_{test} is the number of test points used to evaluate the prediction error.

It is useful in practice to define a *normalized root mean squared error*, $NRMSE$, for example in terms of the maximum value for $y = f(\mathbf{x})$ in the test set

$$NRMSE = 100\% \times \frac{RMSE}{\max\left(y_i \mid_{i=1}^{N_{test}}\right)} \; . \tag{3}$$

Then it is possible to remove magnitudes and make comparisons on the accuracy of different models.

The squared *sample linear correlation coefficient*, R^2, is another way to measure the quality of fit for approximate models, in which R is given by

$$R\left(X_i, Y_i\right) = \frac{\sum\limits_{i=1}^{N} (X_i - \bar{X})(Y_i - \bar{Y})}{\left[\sum\limits_{i=1}^{N} (X_i - \bar{X})^2 \sum\limits_{i=1}^{N} (Y_i - \bar{Y})^2\right]^{\frac{1}{2}}}, \tag{4}$$

with $\bar{X} = \dfrac{1}{N} \sum\limits_{i=1}^{N} X_i$, for any two random vectors X_i and Y_i, of size N.

Another common strategy, known as *cross-validation*, is defined as follows. For $i = 1, \cdots, N$, let $\hat{y}_{-i}$ denotes the metamodel constructed based on excluding the i-th sampling point $(\mathbf{x}_i, y_i)$ from the original set (*leave-one-out*). Therefore, the cross validation score or PRESS (PREdiction Sum of Squares) is given by

$$CV_N = \frac{1}{N} \sum_{i=1}^{N} \left\{ y(\mathbf{x}_i) - \hat{y}_{-i}(\mathbf{x}_i) \right\}^2 . \tag{5}$$

In order to reduce computational cost in the cross-validation process, the set of sampling points N can be divided into k groups of same size, and the procedure is calculated by removing all points in each k-th group, instead of one single point by time. Therefore, there is a trade-off between accuracy and computational cost to estimate PRESS via this k-fold cross-validation. Meckesheimer et al (2002) suggested $k = \sqrt{N}$ or $k = \frac{1}{10}N$ in order to balance accuracy and efficiency.

3.3 Metamodel Based Optimization

In summary, the metamodel based optimization can be stated as a standard optimization problem, by replacing the true objective function $y = f(\mathbf{x})$ and the n_c constraints $g_i(\mathbf{x})$, that are complex and costly to compute, by their respective fast and cheap surrogates, i.e.,

$$\mathbf{x}_{opt} : \begin{cases} \min\limits_{\mathbf{x}} & \hat{y}(\mathbf{x}) \\ & \\ \text{subjected to (s.t.),} & \hat{g}_i(\mathbf{x}) - g^i_{max} \quad \leq \; 0, \quad i = 1 \ldots n_c \quad , \\ & \\ & \mathbf{x}_{lb} \leq \mathbf{x} \leq \mathbf{x}_{ub} \end{cases} \tag{6}$$

where g^i_{max} is the reference value (target) for i-th constraint[5] and $\mathbf{x}_{lb}$ and $\mathbf{x}_{ub}$ the bounds on the design space. The optimum point $\mathbf{x}_{opt}$ can be found by any available optimization algorithm: gradient based, direct search, genetic algorithm, etc. See in Fig. 13 the basic steps of the metamodel based optimization process. Refer to Queipo et al (2005) and Forrester and Keane (2009) for details.

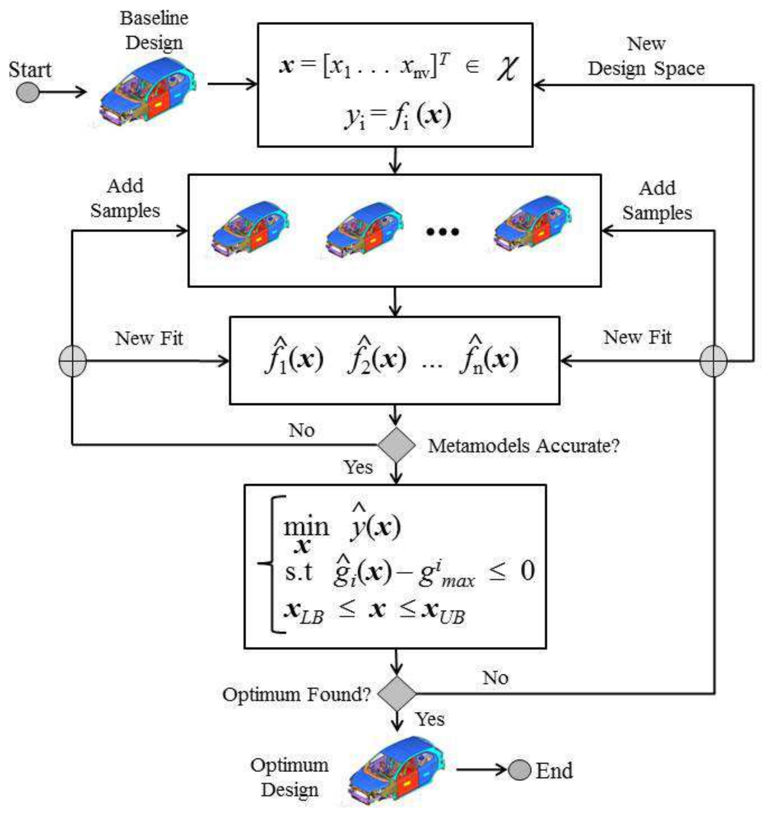

Start
Baseline Design
$\boldsymbol{x} = [x_1 \ldots x_{nv}]^T \in \chi$
$y_i = f_i(\boldsymbol{x})$
New Design Space
Add Samples
Add Samples
New Fit
$\hat{f}_1(\boldsymbol{x}) \quad \hat{f}_2(\boldsymbol{x}) \ldots \hat{f}_n(\boldsymbol{x})$
New Fit
No
Metamodels Accurate?
Yes
$\min_{\boldsymbol{x}} \hat{y}(\boldsymbol{x})$
s.t $\hat{g}_i(\boldsymbol{x}) - g^i_{max} \leq 0$
$\boldsymbol{x}_{LB} \leq \boldsymbol{x} \leq \boldsymbol{x}_{UB}$
Optimum Found?
No
Yes
Optimum Design
End

Figure 13: Metamodel based optimization process. The first phase of the process (first and inner loop) corresponds to creation of the metamodels (as detailed in Fig. 12) to drive the optimization algorithm chosen. The bigger and outer loop is the iterative metamodel based optimization process as whole. At each optimization cycle, if the baseline design is not enough improved, or the global optimum is not found, then at least three paths can be followed: redefine the design space (clustering, windowing, etc.), add more samples to the DOE or improve the quality of fit by changing or tuning the approximation method. The process is nearly the same for multiobjective or multidisciplinary optimization scenarios as well.

[5] Only for notational convenience, without loss of generality, we will assume that all equality constraints $h(\mathbf{x}) = 0$ can be properly transformed into inequality constraints $g(\mathbf{x})$.

3.4 An Illustration with Metamodels

In order to illustrate some metamodeling concepts, let us start by an approximation example with a simple one variable function. As can be observed in Figure 14 both KRG and RBNN match exactly, or almost exactly, all the DOE points (i.e. interpolation models), while PRS and SVR deviate from all the sampling points (i.e. regression models).

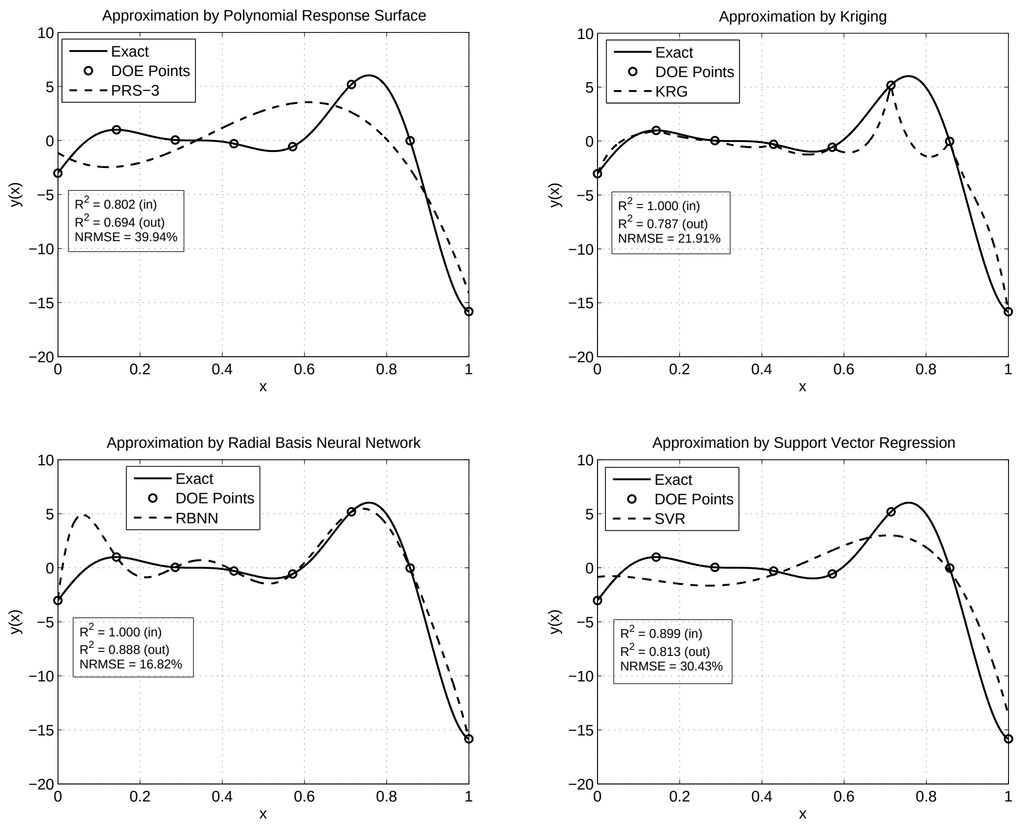


Figure 14: Different metamodels for the same function $y(x) = \frac{3}{10} + \sin\left(\frac{16}{15}x - 1\right) + \sin^2\left(\frac{16}{15}x - 1\right)$, i.e.: PRS, KRG, RBNN and SVR. All models generated by using the SURROGATES Toolbox (see details on Appendix A).

If we need to interpolate the points in the sample (in) and also have good accuracy out of the sample (out), both KRG and RBNN should present problems. For instance, KRG is good in the range $(0 - 0.6)$, but it is very poor in the range $(0.6 - 1.0)$. By its turn, RBNN is good in the interval $(0.4 - 1.0)$ but it overfits the results in the range $(0 - 0.4)$, specially for $x < 0.2$. In other words, for this example KRG and RBNN present good local accuracy which is not true globally.

Alternatively, if the objective is not necessary to interpolate exactly the DOE points, but predict with acceptable accuracy in the whole domain, which means regression, thus SVR or PRS should be considered good for this simple example, since the deviation from the exact values is almost equally spread (or smoothed) along the domain.

Even if we compare the objective error measure numbers, the answer will not be unique as well. That is, the correlation measured by R^2 (Eq. 4) at the data points (in) is perfect for RBNN and KRG, i.e. $R^2 = 1$, which is desired for interpolation. But the R^2 numbers for these two models differ significantly for unknown data (out).

In another way, SRV is quite stable in terms of R^2 for (in) and (out) data, which represents less overfitting, but the global error at points out of sample, measured by $NRMSE$, is higher for SVR as compared to RBNN and KRG.

If we need to consider the computational cost involved to create the approximate models, in addition to the accuracy level, the decision problem becomes more complex. It is known that neural networks, KRG and SVR should by quite slow depending on the number of points and variables involved, since they are based on local optimization loops in the generation and fine tuning of the model parameters. In this sense, PRS is unbeatable and it should be a fairly good solution for this problem presented in Figure 14. Although the error levels of PRS are the worst ones, they are on the same order of SVR and the accuracy can be improved by adding a few more points in the sample, or by increasing the degree of the basis polynomial in the regression model at virtually no cost.

Of course this is a simple illustration problem, but the task of selecting a good or the best model in practice is not easy, specially for higher number of variables and complex nonlinear problems, which becomes more difficult when the cost to increase the sampling space is high, as can be found in engineering design (e.g., automotive crash models, aerodynamic analysis, fluid-structure interaction problems, that run for several hours in high performance computer clusters).

Well, if there is no trivial solution, is there any way to combine a set of distinct approximate models and achieve a merged model with higher accuracy or good accuracy in a broader sense than individual models alone? Is that possible to join the models in such a way that incorporates the advantages and eliminates or alleviates the weaknesses from each model available? Since the computational cost for creating several metamodels is small as compared to the cost to evaluate the true function being approximated, then the combining process or a mixed selection and combining approach starts making sense.

In fact, this question is not new and our objective in the next chapters is to discuss traditional methods developed for combining multiple metamodels for prediction and optimization.

4 Ensemble of Metamodels Background

"All models are wrong; some models are useful."

George E. P. Box, 1919-2013.

In the previous chapter we presented the basic concepts of metamodeling and the most common methods available to create an approximate model for a problem.

As we discussed, the best metamodeling strategy is always dependent on: the degree of nonlinearity of the problem; the number of design variables; the sampling points technique and distribution in the design space; the functional form of surrogate model (shape and tuning parameters) and the error measure adopted.

After a lot of research and comparative studies in this subject performed in the last decades, the conclusion is that it is a problem dependent issue. There is not a unique approximation method that can handle with equal and acceptable accuracy every problem at hand, or at least the majority of problems to approximate. Each method has its pros and cons and the user should be able to choose the best for his own purposes.

As we discussed in Section 3.4, the task for selection of metamodels is not a trivial solution and, since the computational cost for creating several metamodels is small as compared to the cost to evaluate the true function being approximated, the combining process (or a mixed selection and combining approach) can be a feasible way.

This question is not new and our objective in the next sections is to discuss the methods developed for combining multiple metamodels for prediction and optimization.

4.1 Origins and Definitions

Concepts like *ensemble of predictors*, *mixture of experts*, *committee of networks*, etc., are well known in machine learning literature. See for instance Wolpert (1992), Hashem (1993), Perrone and Cooper (1993), Bishop (1995) and Breiman (1996). The idea is that, by the combination of different predictors, it is possible to improve the final model accuracy.

Let us discuss these ideas with a little more detail. Suppose that we have a set of M models: $\hat{y}_1(\mathbf{x})$, $\hat{y}_2(\mathbf{x})$, ..., $\hat{y}_M(\mathbf{x})$, which represent distinct numerical approximations of an output $y(\mathbf{x})$. In addition, all the models have been constructed based on the same set of N sampling points, represented by the design matrix (or input-output matrix)

$$\mathbf{X} = \{(\mathbf{x}_i, y_i)\,,\ i = 1,\ 2, \cdots,\ N\}\,.$$

As discussed previously, what is desired is not a collection of predictors for the same problem, but one single model that represents or predicts as best as possible the true response $y(\mathbf{x})$ for the problem at hand.

The straight forward approach is to generate the collection of predictors and then select the “best one”, in terms of predefined error measures. The best model selected in this way is also known as “naive estimator”, as for instance in terms of mean squared error (MSE)

$$\hat{y}_{Naive}(\mathbf{x}) = \arg\min_{i}\left(MSE\left[\hat{y}_i(\mathbf{x})\right]\right). \tag{7}$$

At first, what is immediately argued is that once the models are generated and the best one is selected from the list, all the effort demanded and information stored in the discarded models are wasted. Second, the generalization properties of the approximation can be compromised since the model with best performance in a validation set can lead to overfitting in future data. Therefore combining properties of different predictors can be a way to add smoothness in the final approximation.

As discussed for instance by Breiman (1996), the idea of combining predictions or forecasts remounts to the late 1960s, with theoretical discussions in statistics and financial fields. Even though, what was not clearly detailed up to early 1990s is *how* to perform such a combination in a formal and systematic way. Up to that point, the techniques for combining predictors were considered as more *art* than *science.* In Wolpert’s own words, many aspects of combining (or stacking) predictors, at that moment, are treated as “black art”, Wolpert (1992).

In this sense, an *ensemble* of models can be defined as a linearly weighted summation[6]

$$\hat{y}_{ens}(w_i\ , \mathbf{x}) = w_0 + \sum_{i=1}^{M} w_i \hat{y}_i(\mathbf{x})\ \ , \tag{8}$$

where $\hat{y}_i(\mathbf{x})$ are the M available distinct models, w_i are the weights associated in the linear combination and w_0 is an intercept term. If $w_0 = 0$ and $w_i = \dfrac{1}{M}$, then it is defined the *simple average* ensemble (SA).

The first developments on finding *optimal weights* w_i in the ensemble are acknowledged to Perrone and Cooper (1993) and Hashem (1993). They devised independently an approach

[6]Most of the publications is focused on linear ensembles, but it can be observed a growth of interest on *nonlinear ensemble methods*, in which any type of approximation should be used to combine the models, e.g., neural networks, support vector regression, etc. See, for instance Yu et al (2005), Lai et al (2006) and Meng and Wu (2012).

that has been presented and discussed in detail in the book by Bishop (1995). In summary, by the minimization of the mean squared error (MSE), with respect to w_i, and with $\sum_i w_i = 1$ as constraint, it follows that

$$w_i = \frac{\sum_{i=1}^{M} \left(\mathbf{C}^{-1}\right)_{ij}}{\sum_{k=1}^{M}\sum_{j=1}^{M} \left(\mathbf{C}^{-1}\right)_{kj}} \;, \tag{9}$$

where the error correlation matrix $\mathbf{C}$ is estimated based on a validation set of N_v sampling points as follows

$$C_{ij} = \frac{1}{N_v} \sum_{n=1}^{N_v} \left[\left(y_n - \hat{y}_i\left(\mathbf{x}_n\right)\right)\left(y_n - \hat{y}_j\left(\mathbf{x}_n\right)\right)\right]. \tag{10}$$

The accuracy of the weights calculated in this way is directly dependent on the quality of the estimation of the matrix $\mathbf{C}$, as in Equation (10). By its turn, the accuracy of $\mathbf{C}$ is dependent on the assumption that the errors $e_i\left(\mathbf{x}\right) = y\left(\mathbf{x}\right) - \hat{y}_i\left(\mathbf{x}\right)$ have zero mean and are uncorrelated, i.e.,

$$E\left[e_i\left(\mathbf{x}\right)\right] = 0 \qquad \text{and} \qquad E\left[e_i\left(\mathbf{x}\right) e_j\left(\mathbf{x}\right)\right] = 0 \;, \qquad \text{if} \quad j \neq i \;, \tag{11}$$

where $E\left[\cdot\right]$ denotes the statistical expectation.

In practice, this condition means that the matrix $\mathbf{C}$ need to be full rank and the N_v sampling points are enough in number and distribution to assure the quality of the estimate of $\mathbf{C}$. Otherwise, the matrix inversion process will become unstable and the estimate of $\mathbf{C}^{-1}$ can be ill-conditioned and therefore unreliable.

If these assumptions hold, it can be demonstrated that (ref. to Perrone and Cooper (1993) or Bishop (1995) for details)

$$MSE\left(\hat{y}_{ens}\right) \leq \frac{1}{M} \sum_{i=1}^{M} MSE\left(\hat{y}_i\right), \tag{12}$$

which means that: (i) the higher the number of distinct models, the lower is expected the MSE for the weighted average ensemble (with a factor of $\frac{1}{M}$); and (ii) $\hat{y}_{ens}\left(\mathbf{x}\right)$ provides the best estimate of $y\left(\mathbf{x}\right)$ in the mean square sense, if the errors $e_i = y_n - \hat{y}_i\left(\mathbf{x}_n\right)$ have zero mean and are uncorrelated.

In practical applications, these theoretical levels of MSE reduction are difficult to achieve because the models $\hat{y}_i\left(\mathbf{x}\right)$ can be highly correlated. In spite of that, we can find in the literature results advocating 10% or higher for reduction in MSE, by using weighted ensemble of predictors.

Motivated by the potential benefits, ensemble methods for prediction still comprise an active area of research that is getting maturity, as can be seen in the recent books dedicated to this topic: Seni and Elder (2010), Zhou (2012) and Zhang and Ma (2012).

Although ensemble methods are well known in machine learning area, specially in predictive statistics and financial forecasting, this approach is relatively new in the engineering field. This is what we will present and discuss with more detail in Section 4.3.

4.2 An Example with Simple Average Ensemble

Before moving forward with the subject, let us apply and discuss the simple averaging ensemble (SA) method in the combination of the four models presented in Section 3.4 and displayed in Fig. 14. The result of this SA ensemble is presented in Fig. 15.

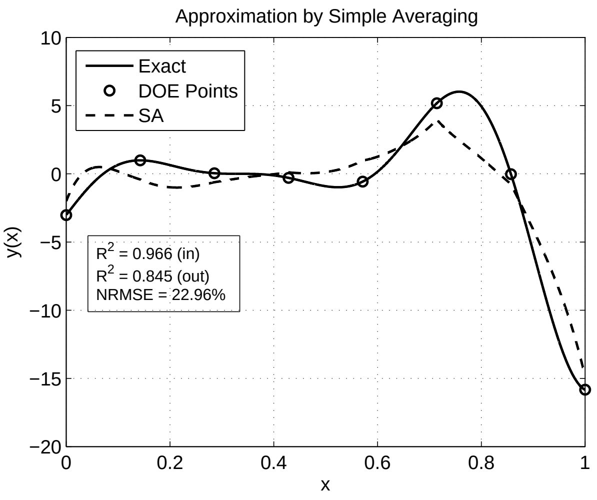


Figure 15: Combining the four metamodels presented in Fig. 14 by simple averaging (SA).

In fact, as can be observed in Fig. 15, the final averaged model (SA) was able to approximate fairly well the function in the whole design domain. As expected, the resulting SA model is only able for regression purposes, since it does not match the DOE points anymore because it tries to respect all the models in the ensemble simultaneously. If we compare the error metrics, R^2 is good for both (in) and (out) data, and the global error measured is $NRMSE = 22.96\%$, which is smaller than the average value of the $NRMSE$ of the previous models individually (27.27%).

Of course this is only a quick toy example and the results cannot be immediately generalized. However, these preliminary results indicate that if we refine the combining approach, there is a good potential to improve the performance of the final ensemble model in the whole design domain.

4.3 Ensemble Methods in Engineering Design

In the review presented by Queipo et al (2005) they remarked that the cost of constructing surrogates (or metamodels) is small compared to the cost of the real computer simulations (i.e., the problems being modeled), thus using multiple surrogates may offer advantages to the use of a single surrogate. In addition they remind that the use of multiple surrogates has a potential to bring several advantages at the optimization and decision-making levels of design phases, specially because: (i) it is expected from a weighted averaged model to provide a prediction with lower variance than any of the individual surrogates; (ii) large variability in the estimated values and variances among surrogates at a point in design space may indicate that the uncertainty at that point is higher than predicted, therefore locations like that are prone to be refined by additional sampling points.

Besides all the controversy, Simpson et al (2008) remark that since the use of multiple surrogates for optimization is affordable when compared to the actual simulations, then the use of a set of surrogates (and possibly a weighted average surrogate) is very appealing in design optimization research and practical applications.

The first application of ensemble methods in engineering design and optimization, inspired by the work in machine learning, is acknowledged to Zerpa et al (2005). They proposed the estimation of the weights w_i in the linear ensemble, Eq. (8), as

$$w_i = \frac{\dfrac{1}{V_i}}{\displaystyle\sum_{j=1}^{M} \frac{1}{V_j}} \ , \tag{13}$$

where V_i is the prediction variance estimation $V\left(\hat{y}\left(\mathbf{x}\right)\right)$ for the i-th metamodel.

Goel et al (2007) proposed heuristic schemes for estimating the weights in a weighted averaged surrogate model (WAS), for instance: PWS (PRESS weighted surrogate), given by

$$w_i^{PWS} = \frac{w_i^*}{\displaystyle\sum_{j=1}^{M} w_j^*} \ , \tag{14}$$

where w_i^* is defined as

$$w_i^* = \left(E_i + \alpha E_{avg}\right)^{\beta} \quad \text{with} \quad E_{avg} = \frac{1}{M}\sum_{j=1}^{M} E_j \ , \tag{15}$$

in which E_i is the global data-based error measure for the i-th surrogate model, in this case PRESS. The parameters $(\alpha < 1$ and $\beta < 0)$ control the importance of averaging and the importance of an individual surrogate respectively and it was suggested $\alpha = 0.05$ and $\beta = -1$.

Acar and Rais-Rohani (2009) estimated the weight factors by solving a direct optimization problem of the form:

$$\min_{w_i} \; E_{rr}\left(\hat{y}_{ens}\ , y\right), \quad \text{subjected to} \; \sum_{j=1}^{M} w_j = 1, \tag{16}$$

and $E_{rr}\left(\hat{y}_{ens}\ , y\right)$ is any selected error metric. In this case it was used RMSE and PRESS to drive the optimization.

Viana et al (2009) defined the optimal weighted surrogate (OWS) by:

$$\min_{w_i} \quad \text{MSE}\left(\hat{y}_{ens}\right) = \mathbf{w}^T\mathbf{C}\mathbf{w}, \quad \text{s.t.} \; \sum_{j=1}^{M} w_j = 1, \tag{17}$$

with the matrix $C_{ij} = \dfrac{1}{N}\tilde{\mathbf{e}}_i^T\tilde{\mathbf{e}}_j$, where $\tilde{\mathbf{e}}$ is the vector of cross-validation errors (PRESS) for the i-th and j-th surrogates. The matrix $\mathbf{C}$ is similar to the one in Eq. (9), but in Eq. (17) it is modified by using cross-validation procedure, i.e., Eq. (5), with the whole set of N sampling points[7].

In all the applications mentioned up to here the weights w_i are considered constant in the design domain ("global" weights). There is no restriction on defining the weights as dependent on the location of the sampling points ("local" weights), i.e., $w_i\left(\mathbf{x}\right)$. Even though, the results published with local weights did not show remarkable improvements, when compared with the constant definition of weights in the design domain. See for example Sanchez et al (2008) and Acar (2010) for more details.

In general, most of the previous research suggested that ensemble of metamodels, or weighted averaged surrogate models (WAS), are able to provide better accuracy than individual metamodels working alone. In other words, it is advocated that weighted averaging schemes should improve the robustness of the predictions and the optimization results, by reducing the impact of poorly fitted surrogates in the ensemble.

On the other hand, as discussed in Viana et al (2009), the computational cost to calculate PRESS can become prohibitive and, in addition, the gains in terms of reduction of RMSE diminishes substantially as the number of variables increases, even with a large number of sampling points. According to this research, unlike established in Eq. (12), none of the ensemble methods tested was able to reduce the RMSE more than 10% when compared to the best model (i.e., the most accurate in terms of PRESS).

In fact, in some examples Viana et al (2009) found that the ensemble model can be less

[7]By considering the general OWS method, Viana et al (2009) defined other variants as follows: OWSfull is the optimal weighted surrogates by using the full $\mathbf{C}$ matrix; OWSconst is OWSfull with an additional positivity constraint for w_j in the optimization (Eq. 17) and OWSdiag is OWSfull modified by using only the diagonal elements of the $\mathbf{C}$ matrix.

accurate than the best model in the set. Therefore, for the problems tested, it was not verified enough evidence that the combination of models is always better than selecting the best model, when PRESS is used as error metric to rank models and drive the estimation of the weights in the ensemble.

Finally, since there are still some controversy and open questions, metamodeling methods for prediction and optimization are included in the list of challenges for research in the next years, as discussed by Yang et al (2013b). Based on what was presented in this chapter, we can conclude that it is still needed some research in this area in order to clarify the open issues and controversy and also stretch the boundaries of applications of multiple surrogates and ensemble methods in design analysis and optimization.

In this sense, our first contribution is on the derivation of an ensemble method based on the concept of least squares approximation. As we will show in the next chapter, this approach is appealing since it is not dependent on any error explicit measure like PRESS, as in most of the previous approaches proposed in the literature and discussed here. In addition, it has a computational cost similar to the simple average, which can be a good feature since some error measures like PRESS can be prohibitive as number of sampling points increase, specially in high dimensions.

5 Ensemble of Metamodels by Least Squares

"When I have clarified and exausted a subject,
then I turn away from it,
in order to go into darkness again..."
J. C. F. Gauss, 1777-1855.

Based on the previous chapter, the main characteristic in the available weighted averaged ensemble schemes is that it is necessary to evaluate an error measure (in general PRESS), which often demands a large number of sampling points in order to be effective. In addition, as it is known, the error computation by cross-validation can be cumbersome, even when a k-fold strategy (refer to Eq. 5) is applied.

The present chapter summarizes the background and main results of the ensemble of metamodels based on least squares. As discussed in Section 1.3, this is the core of the **Phase I** of the research. Further details can be found in our previous work reported in Ferreira and Serpa (2016). This chapter was planned to be as much as possible self contained but, if detailed information is needed, refer to the complete manuscript in the Appendix G.

5.1 Basic Formulation

Here it will be followed an alternative approach for calculating weights in an ensemble model, by minimizing the approximation error, without explicitly calculating any error measure (e.g., PRESS), as discussed in the Section 4.3.

A general linear regression model can be written as

$$y = \beta_0 + \sum_{i=1}^{n_{reg}} \beta_i z_i(\mathbf{x}) + \varepsilon \ , \tag{18}$$

where z_i represents any function $z_i(\mathbf{x})$ of the original *regressors* (or design variables) x_j, and ε is the error of the approximation. See Montgomery et al (2006) for details.

Then, by replacing $z_i(\mathbf{x}) = \hat{y}_i(\mathbf{x})$ and $\beta_i = w_i$, the linear ensemble in Eq. (8) can be rewritten as a linear regression problem, i.e.,

$$\mathbf{y} = \hat{\mathbf{Y}}\mathbf{w} + \boldsymbol{\varepsilon}, \tag{19}$$

where $\mathbf{y} = [y_1 \ \cdots \ y_N]^T$, $\hat{\mathbf{Y}} = [\hat{y}_1(\mathbf{x}_i) \ \cdots \ \hat{y}_M(\mathbf{x}_i)]$ and $\mathbf{w} = [w_1 \ \cdots \ w_M]^T$, for N sampling points and M metamodels.

Therefore, the *standard least squares estimator* for $\mathbf{w}$ in Eq. (19) is given by

$$\hat{\mathbf{w}} = (\hat{\mathbf{Y}}^T\hat{\mathbf{Y}})^{-1}\hat{\mathbf{Y}}^T\mathbf{y}, \tag{20}$$

and, according to the *Gauss-Markov* theorem, if the errors $\boldsymbol{\varepsilon} = [\varepsilon_1 \ \cdots \ \varepsilon_N]^T$ are normally and independently distributed (NID), with zero mean and finite variance, i.e., $\varepsilon_i \in \text{ NID}(0,\sigma^2)$, then $\hat{\mathbf{w}}$ is the referred as the *best linear unbiased estimator* (BLUE) of $\mathbf{w}$. That is, $\hat{\mathbf{w}}$ provides the minimum prediction error, in the least squares sense, and has minimum variance among all unbiased linear estimators. For proofs and details see Montgomery et al (2006) or Björk (1996), with more mathematical and numerical aspects.

The solution of the $N \times M$ set of linear equations in Eq. (20) is well known and can be performed efficiently by standard numerical algebra algorithms. In this way, $\hat{\mathbf{w}}$ minimizes the errors ($e_i = y_i - \hat{y}^i_{ens}$), in the least squares sense, without explicitly calculating any costly error measure, like PRESS for instance.

In addition to the simple formulation and computational efficiency, another interesting property of ensemble methods with optimal weights estimated by Eq. (20) is that they inherit the *least squares variance estimate* $V\left[\hat{y}_{ens}\left(\mathbf{x}\right)\right]$, defined by $\hat{s}^2\left(\mathbf{x}\right)$, for the prediction at each point $\mathbf{x}$, that can be written as

$$\hat{s}^2\left(\mathbf{x}\right) = \hat{\sigma}^2\left[\hat{\mathbf{y}}\left(\mathbf{x}\right)\right]^T\left(\hat{\mathbf{Y}}^T\hat{\mathbf{Y}}\right)^{-1}\hat{\mathbf{y}}\left(\mathbf{x}\right)\ , \tag{21}$$

with $\hat{\mathbf{y}}\left(\mathbf{x}\right) = \left[\hat{y}_1\left(\mathbf{x}\right)\ \hat{y}_2\left(\mathbf{x}\right)\ \cdots\ \hat{y}_M\left(\mathbf{x}\right)\right]^T$, and

$$\hat{\sigma}^2 = \frac{\hat{\mathbf{y}}_{ens}^T\hat{\mathbf{y}}_{ens} - \hat{\mathbf{w}}^T\hat{\mathbf{Y}}\hat{\mathbf{y}}_{ens}}{N - n_v} \tag{22}$$

where $\hat{\mathbf{y}}_{ens} = \left[\hat{y}_{ens}\left(\mathbf{x}_1\right)\ \hat{y}_{ens}\left(\mathbf{x}_2\right)\ \cdots\ \hat{y}_{ens}\left(\mathbf{x}_N\right)\right]^T$.

As we mentioned in Section 2.2.3, by means of the variance estimate, Eq. (21), it is possible to derive an expected improvement function, which is the main ingredient for the application of efficient global optimization algorithms. This branch of the research is covered in the work Ferreira and Serpa (SMO-15-0339), ref. to Appendix G, and it will summarized in Section 7.

It is worth noting that the idea of using least squares regression to find the optimal ensemble weights in fact is not new. As remarked by Hashem (1993), if it is removed the constraint to derive the Eq. (9), the solution for the optimal weights is equivalent to the least squares regression. In spite of that, in this work it was only applied and tested the method based on Eq. (9). In addition, the authors remarked that, due to the intrinsic similarity in the metamodels $\hat{y}_i$, then a "potential problem" of multicollinearity can take place and it can jeopardize the final accuracy of the ensemble predictor. Despite this observation, no further suggestion or investigation on methods to solve (or at least reduce) the multicollinearity issue have been addressed in this work or even in the related research by Perrone and Cooper (1993) or Bishop (1995).

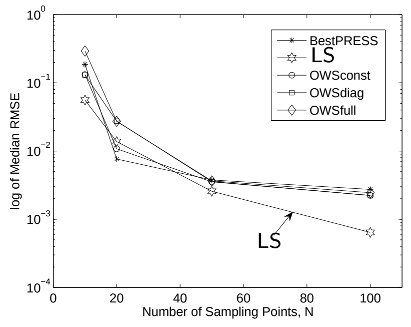
In Breiman (1996) it is presented the concept of *stacked regressions* for variable selection and combination of predictors in CART, *classification analysis and regression trees*, in order to improve accuracy. Breiman discussed that the direct unconstrained minimization of the MSE (i.e. least squares) for the stacked predictors (i.e. ensembles) over the learning set, in general overfits the data and the generalization tends to be poor. In addition, he pointed out that, even using a cross-validation approach to find the weights, the multicollinearity make the ensemble highly sensitive to small variations in the data. In order to handle overfitting and multicollinearity, he suggested and applied the cross-validation allied to ridge regression, i.e., $\sum_i w_i = s$ and constrained minimization, i.e $w_i \geq 0$ and concluded that the later one consistently provided good results for the stacked predictors, by reducing the MSE by 10% against the "best predictor" alone, for two statistical data sets tested.

In the beginning of the present thesis research, we performed some preliminary numerical studies with analytical functions (up to ten variables, $n_v = 10$), in order to test the idea of creating metamodel ensembles by using the least squares approximation concept. Here we will outline the main outcomes of this initial study. If further information is needed, refer to Appendix E for details.

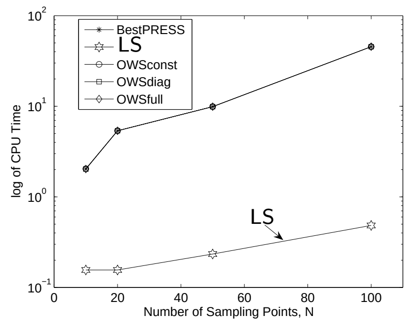
The main results of this preliminary study are compiled in Figure 16. In summary, we observed that:

(i) The accuracy of LS ensemble method is on the same level of the PRESS based ensemble methods for moderate number of sampling points ($N < 50$) and LS was superior only for very dense design spaces, see Fig. 16(a). On the other hand, even for a very large number of sampling points, the computational cost for LS was always lower than OWS variants, i.e., more than one order of magnitude, see Fig. 16(b);

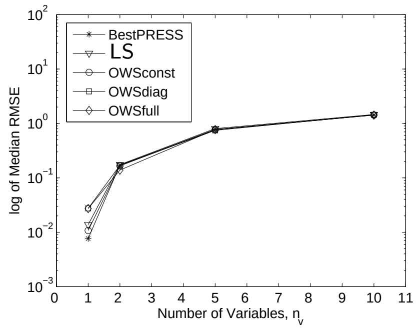
(ii) The accuracy of LS ensemble method is on the same level of the OWS ensemble methods, for increasing the number of variables, see Fig. 16(c). In the same way, the LS method performed much faster than OWS methods, i.e., at least one order of magnitude for low dimension problems (up to 5 variables) and more than two orders of magnitude for high dimension problems (10 variables), see Fig. 16(d);



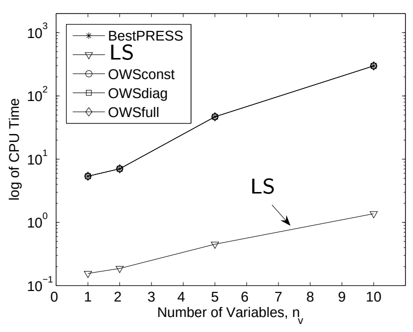
(a) $n_v = 2$



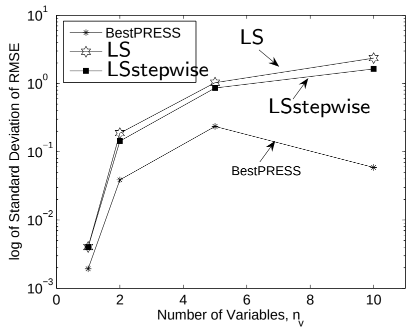
(b) $n_v = 2$



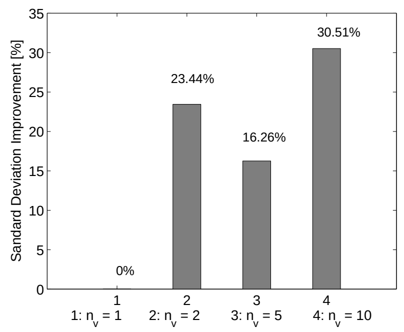
(c)



(d)



(e)



(f)

Figure 16: Main results of a preliminary numerical study with least squares (LS) ensemble in comparison with PRESS based methods, i.e., variations of OWS, Eq. 17.

(iii) On the other hand, LS method presented an undesired instability (measured by the standard deviation of RMSE in 100 runs) as the number of variables increases, see Fig. 16(e);

(v) The variation of accuracy of LS method has been reduced around 30% for 10 variables by applying a *stepwise* selection procedure to the standard least squares solution, in order to reduce the effect of multicollinearity among the metamodels, see Fig. 16(f).

Based on these preliminary results, we concluded that the LS method can be viewed as an alternative to the PRESS based methods, since it is comparable in terms of accuracy and it performs much faster than the other ensemble methods available.

These preliminary results motivated us to a deeper investigation regarding other available variants of least squares methods to handle multicollinearity (e.g., ridge regression, principal components, etc.), in order to understand if it is possible to further improve the accuracy and stability of the LS approach.

In addition, we aimed to understand how the LS ensemble compares to the available ensemble methods developed up to now, summarized in Section 4.3. Although multicollinearity is not clearly discussed in deep on these previous works, it can be observed that in some sense most of them also try to combat the multicollinearity effects in the ensemble, for instance by smoothing the variance of the predictors, as in Eq. (13), or by controlling the *size* of the weights by the constraint $\sum_{j=1}^{M} w_j = 1$, as in Eq. (16) and Eq. (17). On the other hand, these kind of control is not clear in PWS, given by Eq. (14).

As far as we could investigate, there was not found any similar work in the literature in this sense and we aimed to contribute in this front. This preliminary investigation was extended and the results are detailed in Ferreira and Serpa (2016).

5.2 Multicollinearity and Least Squares

The issue of multicollinearity in least squares regression is well known in statistics and related areas. See for instance Montgomery et al (2006), Chapter 11, and the list of references therein for a broad perspective on this subject.

Among the several sources of multicollinearity, the primary ones are: (i) the data collection method (size and distribution of sampling points) an (ii) models with redundant variables (over-defined). During the last decades, several methods have been devised for dealing with multicollinearity in least squares problems. In general, the techniques involve: a) *gathering additional data* and b) *model re-specification*, in order to reduce the prediction errors induced by multicollinearity.

For brevity we will not present here the detailed formulation for all the LS variants. Most of them are covered in Montgomery et al (2006) and details regarding numerical implementation are discussed by Björk (1996). Further information, assumptions and proofs can be found in the references listed in each topic as follows. Refer to Appendix F for a broader discussion regarding multicollinearty and least squares methods and a background on the the main motivations that drive these approaches.

***Stepwise Selection*:** algorithms developed to add and/or delete variables in a regression model to control the accuracy and to reduce sources of error like multicollinearity. In Miller (2002) it is presented an extensive discussion on these methods. We followed the implementation presented in Fang et al (2006) for the variants: F-statistic by Efroymson (1960); Akaike Information Criterion (AIC) by Akaike (1974); Bayesian Information Criterion (BIC) by Schwarz (1978); φ-Criterion by Hannan and Quinn (1979) and Shibata (1984); and the Residual Information Criterion (RIC) by Foster and George (1994);

***Ridge Regression*:** originally published in the two companion papers Hoerl and Kennard (1970a) and Hoerl and Kennard (1970b) to handle nonorthogonal data (i.e., multicollinear). Several algorithms have been proposed in the literature for finding the best ridge estimator, as the *generalized cross-validation* (GCV) procedure by Golub et al (1979), that is one of the most accepted for automation;

***Principal Components Regression*:** a deep and extensive discussion on principal components analysis can be found in Jolliffe (2002). In summary, the original variables x_i are transformed to the principal components (PC) space z_i and then the less significant PC can be identified and removed to reduce the multicollinearity in the system. As in ridge regression, several algorithms have been proposed as well for removing the less significant PC. Golub et al (1979) proposed a variant of GCV suited to principal components selection and it is applied here;

***Constrained and Penalized Least Squares*:** in this cathegory, constraints should be added to the ordinary least squares problem, for example the *sum to unit* and/or the *positivity* of the coefficients, i.e., $\sum\limits_{i=1}^{p} \beta_i = 1$ and $\beta_i \geq 0$ and the problem now is known as *constrained least squares* regression. In order to unify different constrained approaches, a general *penalized least squares* can be defined (ref. Fan and Li (2001) and Fang et al (2006)). The LASSO (least absolute shrinkage and selection) algorithm by Tibshirani (1996) and the SCAD (smoothly clipped absolute deviation) by Fan and Li (2001) are in this category. We followed the implementation presented in Fang et al (2006);

***Generalized and Weighted Least Squares*:** the ordinary least squares equation is modified by adding a weighing matrix, in order to control the variance of the regression coefficients. The most common procedure is the *iteratively re-weighted least squares*, IRLS. We followed the procedure presented in Montgomery et al (2006). See Weisberg (1985) and Amemiya (1985) for different weighing schemes.

***Robust Regression*:** these variants are adopted when the data are not normally distributed or there are possible *outliers*. The literature regarding robust regression is vast and a lot of classes of robust procedures have been proposed (ref. Rousseeuw and Leroy (2003) and Huber and Rochetti (2009)). We adopted the class *M-Estimators* presented in Montgomery et al (2006);

***Total Least Squares*:** if both input and output matrices are contaminated by noise, the ordinary least squares solution can not be accurate (ref. van Huffel and Vandewalle (1991) and Markovsky and van Huffel (2007)). When *total least squares* method is applied, both errors in inputs and outputs are minimized in the Frobenius norm sense. The basic algorithm to solve total least squares is detailed in Fierro and Bunch (1997).

***Effect of Intercept Term*:** The majority of ensemble methods are based on the definition in Eq. (8) and does not take into account the intercept term for generating the linear ensemble of models, i.e., it is assumed $w_0 = 0$. In Hashem (1993), it was remarked the effect of the intercept in the accuracy of the final ensemble of neural networks. They identified that for well-trained network models, the optimal sum of weights tends to one and the constant term tends to zero. However, for poorly trained networks the sum of optimal weights is far from one and the constant term can be significantly different from zero. In linear regression analysis this effect of intercept term is also known. As discussed by Montgomery et al (2006), the effect of intercept can be significant, specially when data lie in a region far from the origin of the design space. Therefore, there is no reason *a priori* to neglect the intercept term in the ensemble models and it can be a good practice to check whether the intercept term improves the accuracy of approximation or not.

5.3 The Augmented Least Squares Approach

In this section we present an approach based on the idea of gathering additional data to reduce multicollinearity. In Fig. 17 we have the true response $y(x)$ (solid line), and four different metamodels $\hat{y}_i(x)$ (dashed lines). It can be observed that the prediction at the $N = 8$ sampling points (circles), used to generate the approximations, is similar for all metamodels, specially in case of interpolating ones, in which the prediction is the same by definition. On the other hand, the prediction is much more different in the additional, $N_{add} = 7$, out-of-sample points (stars), which were not used to generate the metamodels.

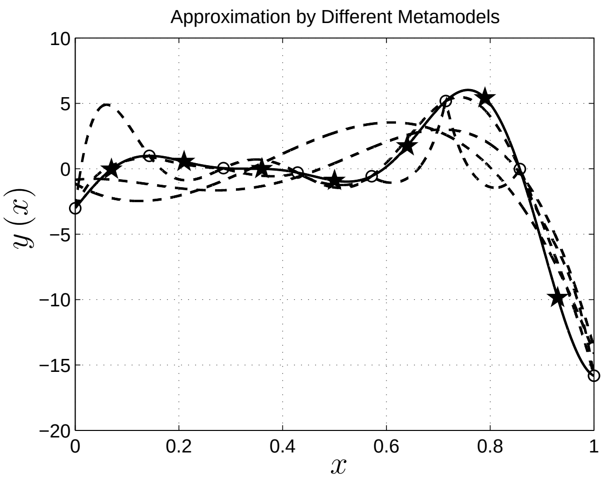


Figure 17: Comparison of different approximation methods for the same problem. Continuous line: true response $y(x) = -(6x-2)^2 \sin(12x-4)$; Dashed lines: metamodels by PRS, KRG, NN and SVR; Circles: sampling points used to create the metamodels; Stars: midpoints, out-of-sample data not used in the approximations. Note that the models tend to be similar near to the sampling points (circles) but they can differ a lot in the non sampled points (stars).

Let us consider two different cases. In the first one, the N points are used to generate the metamodels $\hat{y}_i(x)$ and then the matrix $\hat{\mathbf{Y}} = [\hat{y}_1(x_i) \ \cdots \ \hat{y}_M(x_i)]$ is assembled. In the second case, the metamodels $\hat{y}_i(x)$ are evaluated at the N_{add} points, and an *augmented* matrix $\hat{\mathbf{Y}}_{aug}$ is assembled as follows

$$\hat{\mathbf{Y}}_{aug} = \left[\begin{array}{ccc} \hat{y}_1(\mathbf{x}_1) & \cdots & \hat{y}_M(\mathbf{x}_1) \\ \vdots & \ddots & \vdots \\ \hat{y}_1(\mathbf{x}_N) & \cdots & \hat{y}_M(\mathbf{x}_N) \\ \hline \hat{y}_1(\mathbf{x}_{N+1}) & \cdots & \hat{y}_M(\mathbf{x}_{N+1}) \\ \vdots & \ddots & \vdots \\ \hat{y}_1(\mathbf{x}_{N+N_{add}}) & \cdots & \hat{y}_M(\mathbf{x}_{N+N_{add}}) \end{array}\right].$$

Since the predictions in the augmenting N_{add} points are expected to differ among the metamodels (as in the star points in Fig. 17), it is reasonable to assume that any collinearity among the columns of $\hat{\mathbf{Y}}$ has a good chance to be reduced in the augmented case $\hat{\mathbf{Y}}_{aug}$, if we have enough N_{add} points to take advantage of the diversity in the metamodel predictions in points out-of-sample.

If this conjecture is valid, the ensemble weights $\mathbf{w}$ can be estimated for the augmented system as

$$\hat{\mathbf{w}}_{aug} = \left(\hat{\mathbf{Y}}_{aug}^{T}\hat{\mathbf{Y}}_{aug}\right)^{-1}\hat{\mathbf{Y}}_{aug}^{T}\mathbf{y}_{aug}, \tag{23}$$

with $\mathbf{y}_{aug} = \begin{bmatrix} y_1 & \cdots & y_N \mid y_{N+1} & \cdots & y_{N+N_{aug}} \end{bmatrix}^T$. Since the linear dependency in $\hat{\mathbf{Y}}_{aug}$ is expected to be lower than in $\hat{\mathbf{Y}}$, then the accuracy and stability on estimating $\mathbf{w}$ by means of $\hat{\mathbf{w}}_{aug}$ is expected to be improved as well.

In many practical situations of metamodeling in engineering design, gathering additional points should be time consuming, costly or even impossible. In these cases, the whole data set of N sampling points available should be split in two parts, for example $N = N_{tr} + N_{add}$. In the sequence, the metamodels are created based on the *training* data set N_{tr} and the weights are then calculated based on the augmented system, Eq. (23), with all the N points available. Therefore, in this case it will arise a trade-off regarding the loss of accuracy in the metamodels approximated with less points (i.e., $N_{tr} < N$) and the possible gains of stability with the augmented approach. In this sense, the optimal number of augmenting points, or the ideal rate $\eta_{aug} = \dfrac{N_{add}}{N_{tr} + N_{add}}$, to balance accuracy and stability in the least squares ensemble needs to be investigated.

In summary, the augmented least squares approach can be outlined in steps as follows:

i. Collect $N = N_{tr} + N_{add}$ random sampling points and evaluate the true response y for all points collected;

ii. Generate the approximation models $\hat{y}_i$ by using only the N_{tr} sampling points;

iii. Evaluate the metamodels $\hat{y}_i$ at all $N = N_{tr} + N_{add}$ points and mount the augmented matrix $\hat{\mathbf{Y}}_{aug}$ and the output vector $\hat{\mathbf{y}}_{aug}$;

iv. Solve the system for the weights $\hat{\mathbf{w}}_{aug}$ with Eq. (23).

Let us check this assumption based on the example of Figure 17. We will use the pairwise correlation matrix $\mathbf{R}$ (see Eq. 48 in App. F). Then, by using first only the N sampling points (i.e. circles in Figure 17)

$$\mathbf{R} = \begin{bmatrix} 1.00 & 0.80 & 0.80 & 0.92 \\ 0.80 & 1.00 & 1.00 & 0.92 \\ 0.80 & 1.00 & 1.00 & 0.92 \\ 0.92 & 0.92 & 0.92 & 1.00 \end{bmatrix}$$

and by using one additional point, where the difference in model is highest ($x = 0.79$), and all the seven additional points (i.e stars in Figure 17), then we found respectively

$$\mathbf{R}^1_{aug} = \begin{bmatrix} 1.00 & 0.78 & 0.79 & 0.91 \\ 0.78 & 1.00 & 0.90 & 0.92 \\ 0.79 & 0.90 & 1.00 & 0.87 \\ 0.91 & 0.92 & 0.87 & 1.00 \end{bmatrix} \quad \text{and} \quad \mathbf{R}^7_{aug} = \begin{bmatrix} 1.00 & 0.65 & 0.69 & 0.89 \\ 0.65 & 1.00 & 0.85 & 0.83 \\ 0.69 & 0.85 & 1.00 & 0.81 \\ 0.89 & 0.83 & 0.81 & 1.00 \end{bmatrix} .$$

As can be observed in this example, $\mathbf{R}$ presents high pairwise correlation among models ($R_{ij} > 0.8$) and in addition models 2 (RBNN) and 3 (KRG) are perfect correlated ($R_{23} = 1.00$), when using only the sampling points. On the other hand, as expected, by augmenting the system the correlation is reduced even by using only one extra point, as displayed in $\mathbf{R}^1_{aug}$. Further, it is also observed that the reduction of the correlation is more pronounced by augmenting the system with more points, as shown in $\mathbf{R}^7_{aug}$. Note that, for instance, with 1 extra point $R_{23} = 0.9$ and with 7 extra points $R_{23} = 0.85$; and in case of models 1 and 2, $R_{12} = 0.78$ for 1 extra point, and with 7 extra points $R_{12} = 0.65$.

In this way, with this simple example we can see the potential of this method to reduce multicollinearity in the ensemble method by least squares. In this sense, it is necessary to better investigate the behavior of this method in a broader scenario and also to define how many points are needed to use in order to augment the system and achieve the desired accuracy.

In the Section 5.4, we will illustrate the behavior of the augmented least squares approach, as compared to other LS variants. In Section 5.5 we will summarize the results of Ferreira and Serpa (2016).

5.4 An Example with Least Squares Variants

Once again let us make use of the simple example of the four models presented in Section 3.4 to illustrate the behavior of different least squares variants in the estimation of the weights for the ensemble model. For convenience and easy reference the models presented in Fig. 14 will be repeated here in Fig. 18

As can be seen from the results in Table 1 and Figure 19, the different methods resulted in numerically different weights and final accuracy for the approximation. In case of LSQ, LSQstep and LSQrdg, the weight and errors are technically the same, when compared up to 4 decimal figures. The main difference is found for LSQaug and LSQpc, where the weights vary significantly with respect to the standard least squares.

In addition, it is worth noting that LSQpc resulted in a large error reduction when compared to LSQ and SA, by merging KRG and RBNN equally ($w_2 = w_3 = 0.5$), which are the best two models in the set and by discarding PRS and SVR ($w_1 = w_4 = 0.0$), the poorest ones.

Therefore, it can be inferred from the results of this simple example that using different LS variants can lead to different ensemble models with a potential to improve the quality of fit. We investigated this in detail and the results will be summarized in the next section.

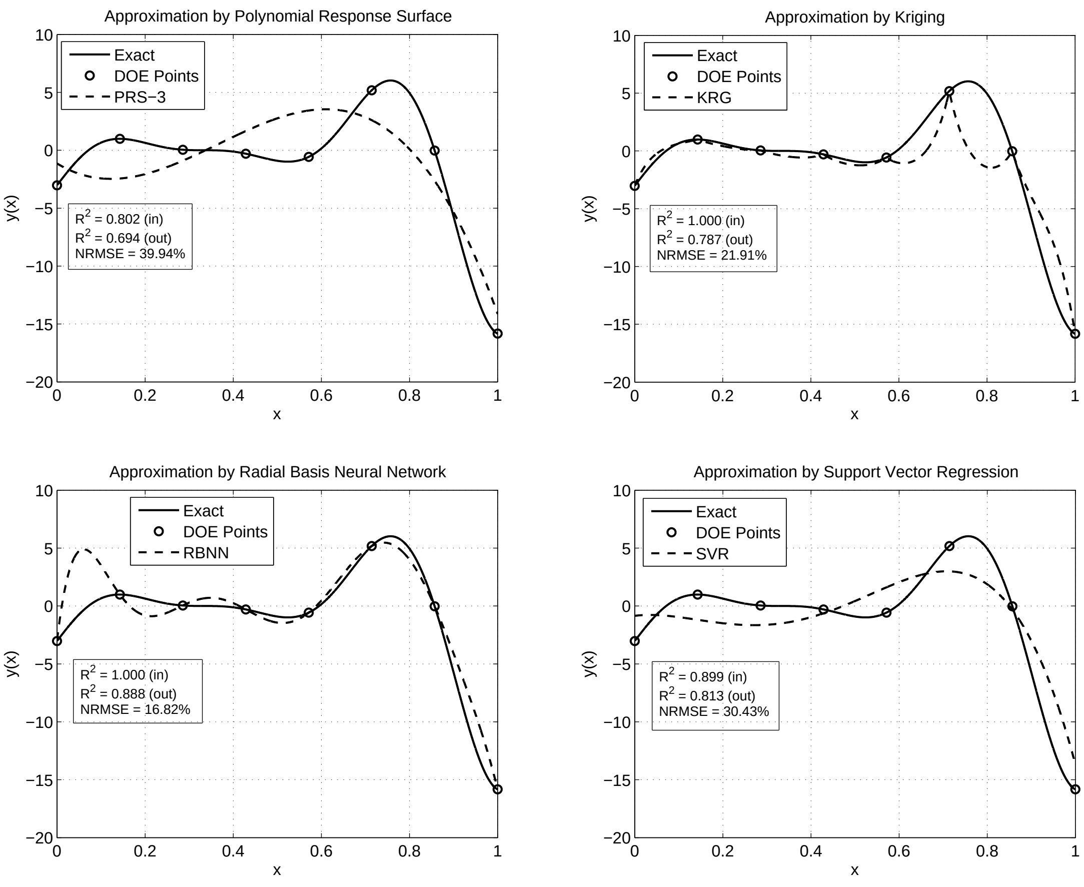
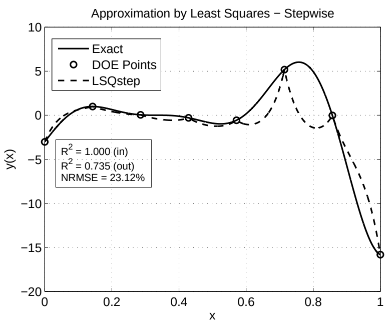


Figure 18: Different metamodels for the same function $y(x) = \frac{3}{10} + \sin\left(\frac{16}{15}x - 1\right) + \sin^2\left(\frac{16}{15}x - 1\right)$, i.e.: PRS, KRG, RBNN and SVR. All models generated by using the SURROGATES Toolbox (see details on Appendix A). The present models are the same displayed in Fig. 14, repeated here for easy reference to the results presented in Tab.1.

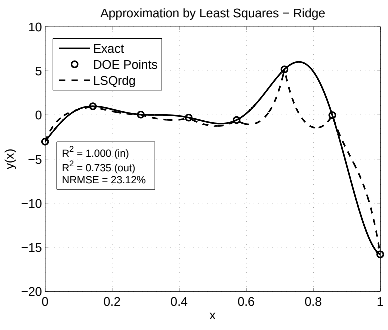
Table 1: Ensemble weights and error measures by different least squares variants, with respect to the models presented in Fig.14. For each method it is displayed the resulting weights w_i, the linear correlation coefficient for points in sample R^2(in) and out of sample R^2(out) and the normalized RMSE.

Method	$w_1 - PRS$	$w_2 - KRG$	$w_3 - RBNN$	$w_4 - SVR$	R^2(in)	R^2(out)	NRMSE
SA	0.250000	0.250000	0.250000	0.250000	0.966	0.845	22.96%
LSQ	0.000000	-0.000014	1.000014	0.000000	1.000	0.735	23.12%
LSQstep	0.000000	0.000000	1.000000	0.000000	1.000	0.735	23.12%
LSQrdg	0.000000	-0.000014	1.000014	0.000000	1.000	0.735	23.12%
LSQpc	0.000000	0.500000	0.500000	0.000000	1.000	0.864	19.07%
LSQaug1	0.000000	1.164681	-0.164681	0.000000	1.000	0.822	19.76%
LSQaug4	-0.207307	0.471877	0.145432	0.673353	0.988	0.888	19.65%
LSQaug7	0.086473	0.609291	0.150702	0.265185	0.989	0.909	16.71%

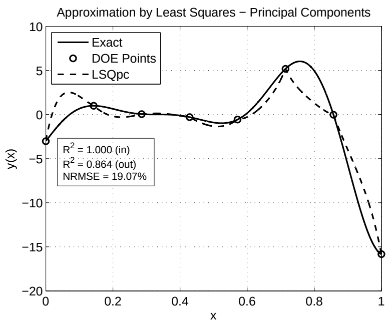
SA: simple average;
LSQ: standard least squares;
LSQstep: least squares with stepwise selection;
LSQrdg: least squares with ridge regularization;
LSQpc: least squares by principal components selection;
LSQaug1: standard least squares with augmented matrix - 1pt;
LSQaug4: standard least squares with augmented matrix - 4pts;
LSQaug7: standard least squares with augmented matrix - 7pts.



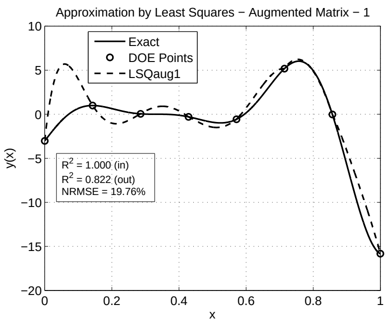
(a) Stepwise



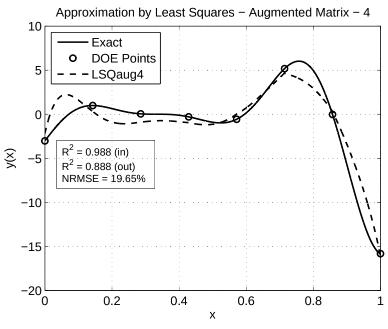
(b) Ridge Regression



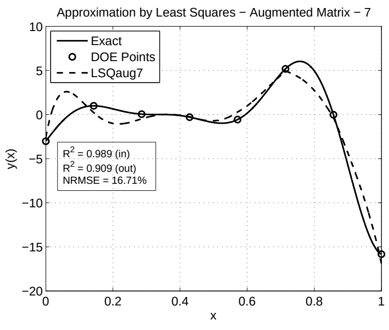
(c) Principal Components



(d) LSQaug1



(e) LSQaug4



(f) LSQaug7

Figure 19: Combining multiple metamodels by using different variants of least squares (i.e., stepwise, ridge regression and principal components) versus the augmented least squares approach (LSQaug) with different number of augmenting points (1, 4 and 7). For this simple example LSQaug performed better than the other methods in terms of accuracy (see details in Tab. 1). Note that by adding more augmenting points the quality of fit improves but it can be seen some saturation between 4 and 7 augmenting points for this case, which suggests that there is an optimal value for η_{aug} for each problem.

5.5 The LS Ensemble Approach: Results Summary

In the work by Ferreira and Serpa (2016) we presented an approach to create ensemble of metamodels (or weighted averaged surrogates) based on least squares (LS) approximation, as discussed in Section 5.3.

The LS approach is appealing since it is possible to estimate the ensemble weights with simple formulation and low computational cost, without using any explicit error metric like PRESS (prediction sum of squares), as in most of the existent ensemble methods published in the literature.

The proposed LS approach is a variation of the standard least squares regression by augmenting the matrix system, Eq. (23), in such a way that reduces the effects of multicollinearity, inherent to calculation of the ensemble weights.

In numerical experiments performed in Ferreira and Serpa (2016) we investigated the number of augmenting points needed by the augmented LS method, in order to be effective. In summary, we observed that for low dimension problems the ideal number of data points to augment the system is around one third (30% to 35% of the data), and in case of high dimension this number is about 10% of the sampling points.

We also investigated the effect of increasing the number of metamodels in the augmented least squares ensemble. In summary, we did not confirm the theoretical predictions that state: "the higher the number of distinct models, the lower is expected the MSE for the weighted average ensemble", as discussed for example in the pioneer studies like Perrone and Cooper (1993) or Bishop (1995) and summarized in Eq. (12).

For low dimension problems we observed a reduction in error from four to eight models in the ensemble and the average error level remained constant up to thirty metamodels. On the other hand, in case of high dimensions, the average error level remained almost constant by increasing the number of metamodels in the ensemble, from four to thirty models. These results suggest that the typical problems of engineering applications do not meet the assumption of uncorrelated errors with zero mean and, in addition the underlying multicollinearity among models compromises the accuracy of the final ensemble prediction.

In the sequence, we tested and compared the augmented LS approach with different LS variants (i.e., stepwise selection, ridge regression, principal components, constrained least squares, weighted and total least squares) and also with the other existent ensemble methods (Section 4.3), by means of analytical benchmark functions and real-world applications, in problems in the range of two to forty-four variables.

In comparison with existent LS variants, the augmented LS approach was able to reduce the level of prediction error (average RMSE) and the instability (standard deviation of RMSE) due to multicollinearity. For the test problems investigated and by using 33% of the data as augmenting points, none of the other existent LS variants was able to surpass the performance of the proposed augmented LS approach in terms of prediction error level (average RMSE) and stability (standard deviation of RMSE).

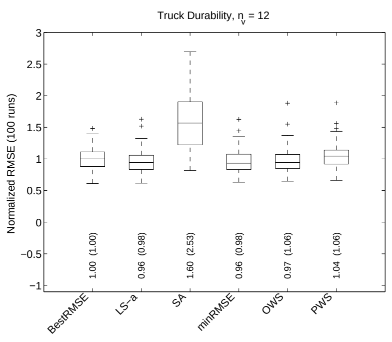
In the cases investigated, it was not observed any improvement in the error levels or variation of LS solution by letting the intercept term, i.e., w_0 in Eq. (8), to be nonzero in the ensemble. Possibly there is a balance in the value of the remaining weights in the ensemble in order to compensate the presence of the intercept in the final model. On the other hand, since the intercept term is not necessarily null, it is recommended to check for each problem and data (DOE) if the error levels can be improved or not by using the intercept term in the ensemble equation. Since in most cases the ensemble calculation is fast, specially when compared to the evaluation of the true models, then this verification should be worthwhile.

When compared with other weighted average ensemble schemes published in the literature (i.e, simple averaging (SA), PWS, direct optimization of RMSE and OWS methods, detailed in Section 4.3), in general the augmented LS approach performed with good accuracy and stability for prediction purposes, in the same level of the existent ensemble methods. See boxplots[8] in Fig. 20 (a), (b) and (c).

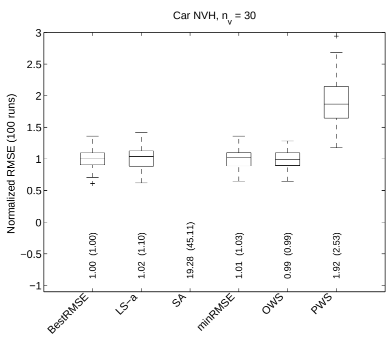
This trend was observed for both analytical and "real-world" engineering problems (see Appendix D.2). In terms of computational cost, for the studied problems the LS approach performed up to three orders of magnitude faster than the PRESS based methods, and in general it is comparable in terms of computational cost to the fastest method (i.e., the simple averaging ensemble, SA). See these results with engineering problems in Fig. 20 (c), (d) and (f).

As we discussed previously, an additional feature (nonexistent to the other existent ensemble methods) is that the ensemble of metamodels based on least squares has a variance estimate function, that enables the application in the EGO context. The developments and results in this branch of application for the proposed ensemble approach is presented in the work Ferreira and Serpa (SMO-15-0339), ref to Appendix G, and will be summarized in Chapter 7.

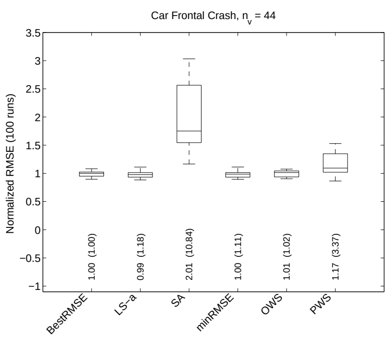
[8]The definition of boxplot is in Appendix B.



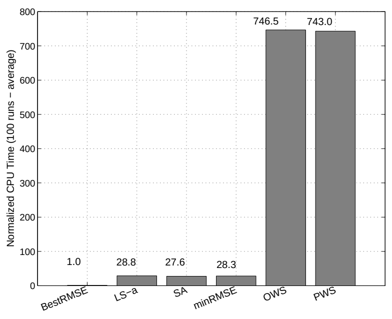
(a) Truck Durability, $n_v = 12$



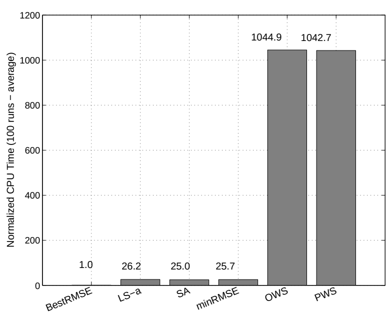
(b) Car NVH, $n_v = 30$



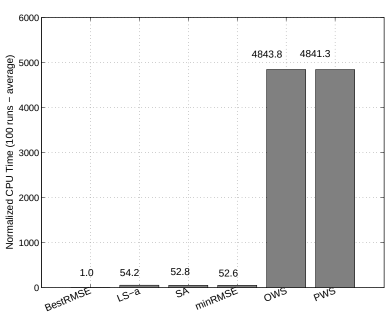
(c) Car Frontal Crash, $n_v = 44$



(d) Truck Durability, $n_v = 12$



(e) Car NVH, $n_v = 30$



(f) Car Frontal Crash, $n_v = 44$

Figure 20: Comparison of accuracy and computational time among the augmented least squares (LS-a, $\eta_{aug} = 33\%$) and different ensemble methods for engineering test problems with $n_v = 12$, $n_v = 30$ and $n_v = 44$ variables. The accuracy of LS-a (measured by RMSE) is in the same level of the most accurate competitor methods, see boxplots on Figs. (a), (b) and (c). On the other hand the computational cost of LS-a (time) is in the same level of SA and minRMSE but it is much more faster than the PRESS based methods PWS and OWS, see Figs. (d), (e) and (f). BestRMSE represents the best the model in terms of RMSE for each run; SA is the simple average ensemble, Eq.(15); PWS refer to Eq. (14); minRMSE refer to Eq. (16) and OWS refer to Eq. (17). All values normalized with respect to BestRMSE in each run. In all cases, the number of metamodels in the ensemble is $M = 4$. Further details regarding these numerical experiments can be found in Ferreira and Serpa (2016).

6 Efficient Global Optimization (EGO)

"The optimal is the enemy of the good."
Voltaire, 1694-1778.

As pointed out by Forrester and Keane (2009): "the Holy Grail of global optimization is finding the correct balance between exploitation and exploration". Algorithms that favor exploitation (search of the optimum) can be quite slow to converge and in addition be trapped at a local extrema points and not be able to reach the global optimum. On the other hand, pure exploration (improvement of the surrogate and search in the hole design space) may lead to a waste of resources (function evaluations, simulations).

At the end of the day, the persistent dilemma for metamodel based optimization is: "*how to generate an accurate metamodel for prediction (exploration) and also to find a reasonable optimum of the objective function (exploitation), at a minimum cost, i.e., with a minimum number of sampling points (or evaluations of the true function)?*"

As an attempt to solve this dilemma, after the work of Schonlau (1997) and Jones et al (1998), efficient global algorithms (EGO) emerged as feasible tools to balance exploitation and exploration of the design space.

In this sense, instead of using a "one-stage approach" (i.e., generate as much as possible sampling points and evaluate them at once), researches on sequential sampling approaches have been performed during the years. The main objective is to find a systematic way to start the process with a small DOE and increase iteratively the number of sampling points, based on some criteria of quality of fit, uniformity or improvement in direction of the global or local optima. There are good reviews on this subject, see for instance: Jones (2001), Jin et al (2002), Sóbester et al (2004), Sóbester et al (2005), Viana et al (2010).

There is still some controversy regarding the effectiveness of sequential sampling versus one-stage approaches. Refer to a detailed discussion in Jin et al (2002) and a more recent panorama in Viana et al (2010). On the other hand, it can be observed an increasing research interest on efficient global optimization (EGO) approaches published recently, as for example Henkenjohann and Kukert (2007), Ponweiser et al (2008), Ginsbourger et al (2010) and Viana et al (2013).

The focus of our work here is to extend the concepts of EGO algorithms by using the concept of LS ensembles. Therefore we present some background on EGO method in the next sections.

6.1 Basic Formulation of EGO

Since the sampling data set χ is arbitrary, the determination of $\hat{y}(\mathbf{x})$ can be stated as "a realization of a stochastic process". In this sense, the approximation can be modeled as Gaussian, i.e., a normally distributed random variable $\hat{Y}(\mathbf{x})$, with mean $\hat{y}(\mathbf{x})$ and variance $\hat{s}^2(\mathbf{x})$.

Efficient global optimization algorithms are centered in the concept of maximum expected improvement. Let $y_{min} = \min(y_1, \ldots, y_n)$ be the best current value for the function $y(\mathbf{x})$ in the sampling data set. Then, as described in Forrester et al (2008), the probability of an improvement $I(\mathbf{x}) = y_{min} - Y(\mathbf{x})$ of a point $\mathbf{x}$ with respect to y_{min}, can be calculated as

$$P[I(\mathbf{x})] = \frac{1}{\hat{s}\sqrt{2\pi}} \int_{-\infty}^{0} \exp\left(\frac{-\left[I - \hat{y}\right]^2}{2\hat{s}^2} \right) \mathrm{d}I, \tag{24}$$

by abbreviating the dependence of I, $\hat{s}$ and $\hat{y}$ on $\mathbf{x}$.

On the other hand, the amount of improvement expected can be obtained by taking the expectation $E[\cdot]$ of $\max(y_{min} - Y(\mathbf{x}), 0)$, which leads to

$$E[I(\mathbf{x})] = I(\mathbf{x})\mathbf{\Phi}\left(\frac{I(\mathbf{x})}{\hat{s}(\mathbf{x})}\right) + \hat{s}(\mathbf{x})\phi\left(\frac{I(\mathbf{x})}{\hat{s}(\mathbf{x})}\right), \tag{25}$$

for $\hat{s}(\mathbf{x}) > 0$, and $E[I(\mathbf{x})] = 0$, otherwise. In this equation, $I(\mathbf{x}) \equiv (y_{min} - \hat{y}(\mathbf{x}))$, where $\hat{y}$ is a realization of Y, and $\mathbf{\Phi}(\cdot)$ and $\phi(\cdot)$ are respectively the normal cumulative distribution and normal probability density functions.

Eq. (25) is known as the *expected improvement function* of any point $\mathbf{x}$ in the design space with respect to the current best value y_{min}. For details in the derivation see Forrester and Keane (2009) or the original publications of Schonlau (1997) and Jones et al (1998).

By means of Eq. (25), the EGO algorithm can be defined for any metamodel $\hat{y}(\mathbf{x})$ that provides an uncertainty estimate $\hat{s}(\mathbf{x})$. EGO was originally devised and is traditionally applied with KRG (see basic formulation at Appendix C), but other models like PRS, RBF and other Gaussian models can be used as well. See for example Sóbester et al (2004) in which RBF was applied. In addition, as we mentioned previously and detailed in Section 5.1, by means of the variance estimate, Eq. (21), it is possible to derive an expected improvement function for the LS ensemble method as well.

Traditionally, the available EGO algorithms are based on kriging approximation and a single infill point is generated per optimization cycle. In Section 6.2 the standard EGO algorithm will be detailed and in Section 6.4 some possible extensions are discussed. Our EGO approach based on LS ensembles will be detailed in Section 7.

6.2 The Standard EGO Algorithm

The standard EGO algorithm can be summarized in the following steps:

i. Define a set χ of N sampling points and start the optimization cycles ($j \leftarrow 1$);

ii. Evaluate the true response $y(\mathbf{x})$ at all data sites in χ, at the current cycle j, and set

$$y_{min} \leftarrow \min(y_1, \cdots, y_N);$$

iii. Generate the metamodel $\hat{y}(\mathbf{x})$ and estimate $E[I(\mathbf{x})]$ with all the data points available in the sampling space χ, at the current cycle j;

iv. Find the next infill point $\mathbf{x}_{N+1}$ as the maximizer of $E[I(\mathbf{x})]$, $\chi_{infill} \leftarrow \mathbf{x}_{N+1} = \max E\left[I(\mathbf{x})\right]$; evaluate the true function $y\left(\mathbf{x}\right)$ at $\mathbf{x}_{N+1}$ and add this new point to the sampling space: $\chi \leftarrow \chi \cup \chi_{infill}$;

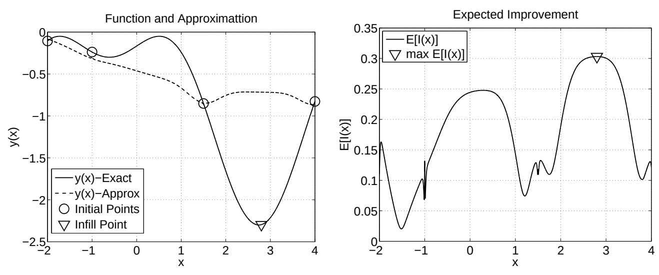
v. If the stopping criteria is not met, set

$$y_{min} \leftarrow \min(y_1, \cdots, y_{N+1}),$$

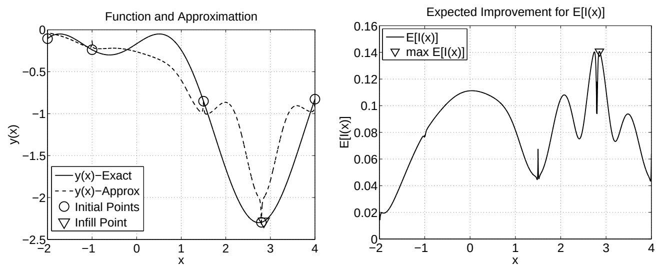
set ($N \leftarrow N + 1$), update cycle counter ($j \leftarrow j + 1$), and return to step iii. Otherwise, finish the EGO algorithm.

It is important to note that, as can be inferred by the definition of the standard EGO algorithm, the optimization process is driven by maximizing the auxiliary expected improvement $E[I(\mathbf{x})]$ function, instead of minimizing the original objective function $y\left(\mathbf{x}\right)$.

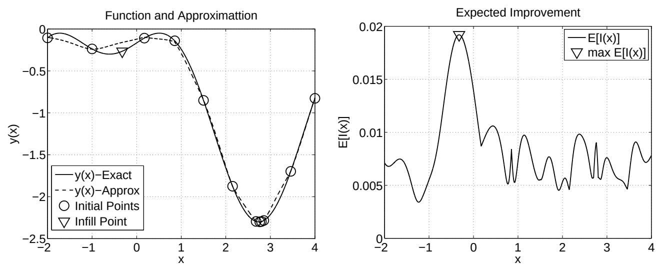
In this case, any valid optimization algorithm available to found the maximum of the expected function at each cycle can be used such as gradient based, non-gradient, genetic, evolutionary, mixed approaches, etc. In the present work we used the Matlab Genetic and Pattern Search built-in algorithms. Further details can be found in Section 8.1.2.



(a) Cicle 01



(b) Cicle 02



(c) Cicle 09

Figure 21: Illustration of the EGO algorithm by using the expected improvement function, $E[I(\mathbf{x})]$. The sampling points (circles) were chosen coarsely in the domain in order to generate a very poor approximation in the begining of cycle 01. At each cycle, the maximum of $E[I(\mathbf{x})]$ (triangle) is the suggested infill point for the next optimization cycle. Note that at cycle 02 the optimum is almost exactly found (exploitation), even with a poor approximation model. After cycle 09, the optimum region is very dense and the metamodel quality is very good in the whole design domain (exploration).

6.3 On Balancing Exploration and Exploitation

As we discussed previously (Section 2.2.2) the aspect of single versus multiple infill points per cycle is known and discussed since the origins of EGO-type algorithms (Schonlau (1997) and Jones et al (1998)).

Observe in Fig. 21 that the expected improved function $E[I(\mathbf{x})]$ can be highly multimodal, i.e., with many local maximum points in the whole design domain. Another trend in the progress of EGO-type algorithms is that infill points can be generated too close during several cycles in sequence, without effective improvement (exploitation) and with no or little contribution for the quality of the metamodel in the other regions of the design space as well (exploration).

Note at cycle 09 in Fig. 21 the high density of infill points generated in the vicinity of the true minimum.

Therefore, if it is possible to control efficiently the number of infill points generated per cycle, then we have a potential to reduce the number of sampling points needed to reach the minimum and also reduce the processing time, since the total number of optimization cycles can be much less than with a single point approach.

Sóbester et al (2004) used a multistart optimization algorithm to find multiple maximum points for the expected improvement function and take advantage of parallel processing resources. Their results indicated accelerated convergence for the optimization, with significant reduction in processing time.

In the last years we can observe an increasing research interest on EGO approaches with multiple infill points per cycle. Henkenjohann and Kukert (2007), Ponweiser et al (2008) and Ginsbourger et al (2010), for instance, proposed different implementations of parallel EGO approaches by extending the concepts of *generalized expected improvement* and *m-step improvement* proposed in the work by Schonlau (1997).

In a different way, Viana et al (2013) proposed the MSEGO (multiple surrogates EGO). In this case, they used multiple surrogates simultaneously at each cycle of EGO algorithm. With at least one kriging model available, they devised a way to transfer (or export) the uncertainty estimate for this model to the other non-kriging models in the set. By means of different uncertainty estimates, they generate different instances of expected improvement functions to be maximized and to provide multiple parallel infill points in each EGO cycle.

Forrester and Keane (2009) presented and discussed in detail other infill criteria developed in order to try a good balance between exploitation and exploration, as for example: *Goal Seeking* and *Conditional Lower Bound*. It is pointed out that choosing a suitable convergence criterion to determine the end of the infill process and terminate the optimization cycles can be rather subjective. The equilibrium between exploration and exploitation is problem dependent and the perfect balance can be considered "utopia", as remarked by Forrester and Keane (2009).

Although it is possible to see some advance on devising targets or stoping criteria for metamodel based optimization, see for instance Queipo et al (2013), in practice there is still no clear definition or a systematic way to declare the convergence of EGO-type algorithms. In the engineering application, the metamodel based optimization process is stopped when the computer resources, costs or schedule are not available anymore in order to continue the design optimization process.

As remarked by Forrester and Keane (2009), the metamodel based optimization must always include some form of iterative search and repetitive infill process to ensure the accuracy in the areas of interest in the design space, as for instance Ge et al (2015), and this is in line with the previous discussion in Section 2.2.1 and later displayed in detail at Fig. 13.

6.4 Extensions for the EGO algorithm

The EGO algorithm can be extended to handle constraints in the optimization by using the concept of probability of improvement. The basis for this extension can be found in Schonlau (1997) and with details and applications in Forrester et al (2008) and Han and Zhang (2012).

By following the derivation presented in Han and Zhang (2012), the idea is to find the probability of satisfying the constraints $g_i(\mathbf{x})$. In other words, when $P[G_i(\mathbf{x}) \leq 0] \to 1$, the constraint is satisfied; otherwise, when $P[G_i(\mathbf{x}) \leq 0] \to 0$, the constraint is violated. Analogously to Eq. (24), $P[G_i(\mathbf{x}) \leq 0]$ can be calculated as

$$P[G_i(\mathbf{x}) \leq 0] = \frac{1}{\hat{s}_i\sqrt{2\pi}} \int_{-\infty}^{0} \exp\left(-\frac{[G_i - \hat{g}_i]^2}{2\hat{s}_i^2}\right) \mathrm{d}G_i \tag{26}$$

where $G_i(\mathbf{x})$ is the random variable related to $\hat{g}_i(\mathbf{x})$ and $\hat{s}_i(\mathbf{x})$ is the respective constraint uncertainty estimate.

In this way, the step (iv) of the EGO standard algorithm presented at the end of Section 6.2 can be modified as follows in order to accommodate n_c *independent* and *uncorrelated* constraints, i.e.,

$$\mathbf{x}_{N+j} = \begin{cases} \max\limits_{\mathbf{x}} & E[I(\mathbf{x})] \prod\limits_{i=1}^{n_c} P[G_i(\mathbf{x}) \leq 0] \\ \\ \text{s.t.,} & \mathbf{x}_{lb} \leq \mathbf{x} \leq \mathbf{x}_{ub} \end{cases}. \tag{27}$$

We extended the implementations presented in Ferreira and Serpa (SMO-15-0339), ref. to Appendix G, in order to handle constrained optimization by means of Eq. (27). In order to test the effectiveness of our EGO approach in constrained optimization, some numerical experiments were performed with analytical benchmark functions and engineering problems. These numerical tests will be described in Sections 8 and 9.

The treatment of multiobjective optimization problems can be also extended by using the concept of probability and expected improvement. It is required a more elaborated derivation that is out of the scope of this text and the details can be found in Forrester et al (2008).

In other direction, Jurecka (2007) has successfully extended and applied the EGO concept to treat robust optimization problems as well, what is extremely important when the design variables and functions are not deterministic.

As we discussed previously, objective convergence criteria is a demand for metamodel based optimization and EGO-type algorithms as well as, for example, the one proposed by Queipo et al (2013).

In addition, methods for reducing the number of sampling points (true function evaluations) and, at same time, accelerating the overall convergence need to be investigated. In this sense, the derivation of hybrid methods by combining classical sequential sampling or other adaptive approaches (clustering, etc.) with EGO-type algorithms can be a promising front of research.

These and other possible extensions of EGO-type algorithms are of interest for research and practical applications and we intend to explore this field in our future work.

7 LS Ensemble of Metamodels EGO (LSEGO)

"Premature optimization is the root of all evil."
D. E. Knuth, 1938-

As discussed in Section 1.3, this chapter summarizes the main formulations and results achieved in the **Phase II** of the thesis work. The results were submitted for publication in Ferreira and Serpa (SMO-15-0339). In the same way, this chapter was planned to be self contained but, if detailed information is needed, refer to the complete manuscript in the Appendix G.

Let us recall the variance estimate for the LS ensemble defined in Section 5.1 by means of Eq. (21).

Therefore, if $\hat{s}^2(\mathbf{x})$ is a good estimate for the uncertainty of the LS ensemble of metamodel at the design space, then it can be used to derive the expected improvement function (Eq. 25) in order to drive EGO algorithms, by using any kind of metamodels $\hat{y}_i(\mathbf{x})$ and not only kriging.

In fact, we cannot verify or prove *a priori* that $\hat{s}^2(\mathbf{x})$ is good or not for our purposes in the EGO context. Since it is a well-established and accepted estimate for general least squares regression, it will be applied in the present work without any proof. The main assumption is that if the LS ensemble of metamodel has reasonable prediction accuracy, therefore the variance prediction $\hat{s}^2(\mathbf{x})$ should be reasonable as well. Our numerical experiments showed that $\hat{s}^2(\mathbf{x})$ works well for generating the EI function and the optimization is convergent in several problems investigated. These results and conclusions will be presented and discussed in the next sections.

On the other hand, as mentioned by Viana et al (2013), although there can exist several measures of quality for the variance estimate, it is possible to infer the behavior by using the coefficient of linear correlation $\rho(|e((x))|, \hat{s}((x)))$, where $e(((x))) = y((x)) - \hat{y}((x))$ is the actual prediction error between the exact function and the approximation, in this case a LS-a ensemble. We will use the same approach here by means of a simple illustration presented in Fig. 22.

In this case, for a coarse sampling plan the quality of fit for LS-a is very poor as can be observed in Fig. 22. On the other hand, the expected improvement function is able to suggest infill points ($max\ E[I((x))]$) where the variance is high for the approximation. The correlation in this case is very good, i.e., $\rho(|e((x))|, \hat{s}((x))) = 0.85$ with several points close to the 45 degrees line, and it is possible to observe that $\hat{s}((x))$ plays a good job estimating the behavior of the actual prediction error $e((x))$ through the design space.

As we observed in preliminary numerical experiments, as the optimization cycles evolve the quality of fit for the metamodel is naturally improved by the infill points added. As consequence the quality of the estimation of $\hat{s}((x))$ also increases and the convergence for LSEGO process is favored. In the next sections we will present and discuss the behavior of LSEGO for different benchmark problems.

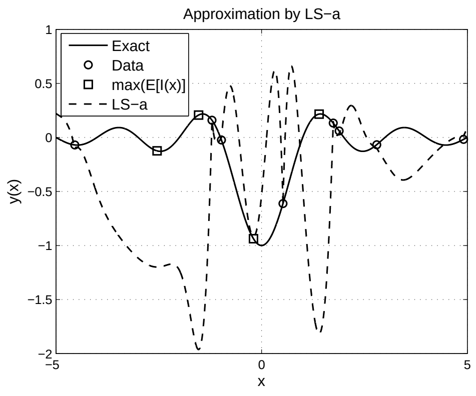
As presented in the review by Haftka et al (2016), different classes of algorithms or strategies can be defined with the objective of balancing exploitation and exploration of the design space during the optimization. It is still an open question and active area of research to answer the question of how to add multiple infill points simultaneously in an efficient way.

Sóbester et al (2004) used a multistart optimization algorithm to find multiple maximum points for the expected improvement function and take advantage of parallel processing resources. Their results indicated accelerated convergence for the optimization with significant reduction in processing time.

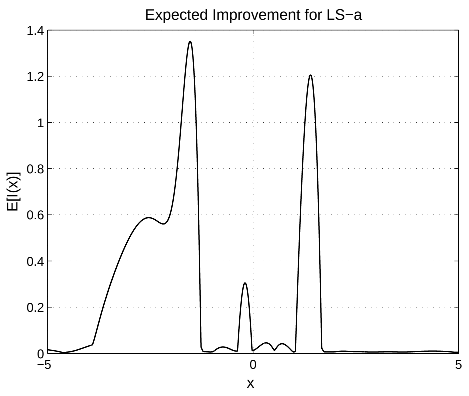
In the last years we can observe an increasing research interest in EGO approaches with multiple infill points per cycle. Henkenjohann and Kukert (2007), Ponweiser et al (2008) and Ginsbourger et al (2010), for instance, proposed different implementations of parallel EGO approaches by extending the concepts of *generalized expected improvement* and *m-step improvement* proposed in the work by Schonlau (1997).

Ginsbourger et al (2010) proposed the multi-points or (q-points) expected improvement (q-EI). They derived an analytical expression for 2-EI and statistical estimates based on Monte-Carlo methods for the general case. Since Monte-Carlo methods can be computationally expensive, they also proposed two classes of heuristic strategies to obtain approximately q-EI-optimal infill points, i.e., the Kriging Believer (KB) and the Constant Liar (CL). Based on numerical experiments, they reported that CL appears to behave as reasonable heuristic optimizer of the q-EI criterion.

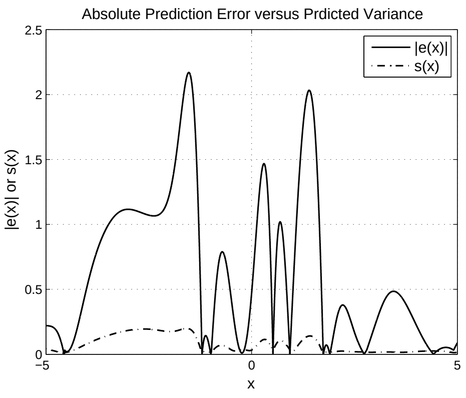
The use of probability of improvement (PI) to generate multiple infill points was discussed by Jones (2001) as a variant of EGO. In this case EGO uses PI beyond a given target as selection criterion. Maximizing PI based on different targets is a way to balance local (exploitation) and global (exploration) searches. Aggressive targets favor more exploitation than exploration and it is not clear how to set these targets properly.



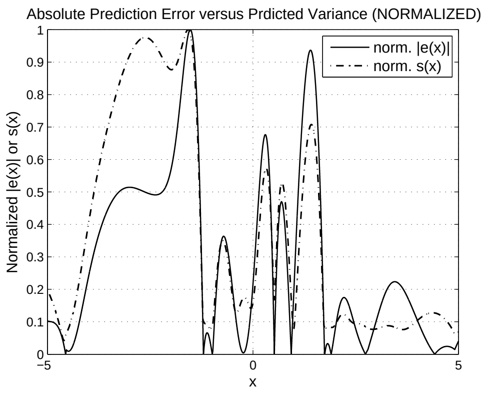
(a)



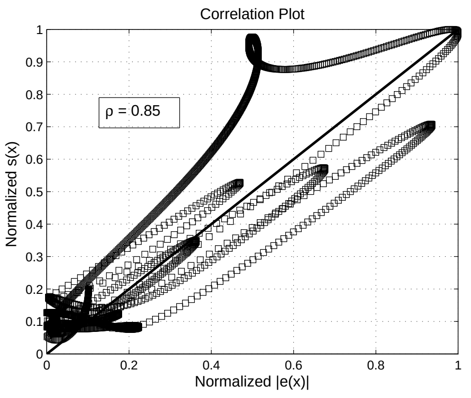
(b)



(c)



(d)



(e)

Figure 22: Correlation between $|e((x))|$ and $\hat{s}((x))$ for the *sinc* function $y(x) = \frac{sin(x)}{x}$. In (a), for a very coarse sampling plan the quality of fit for LS-a is quite poor in the beginning of the LSEGO optimization process. The expected improvement function $E[I((x))]$ in (b) is able to suggest reasonable infill points ($max\ E[I((x))]$), labled as square markers in (a). Observe in (e) that $\hat{s}((x))$ is playing a good job estimating $|e((x))|$ in the design space, with correlation $\rho\left(|e((x))|, \hat{s}(x)\right) = 0.85$ and several points close to the 45 degrees line. Note in (c) and (d) the similarity of $|e((x))|$ and $\hat{s}((x))$. In (d) $|e((x))|$ and $\hat{s}((x))$ were normalized to remove the scale effects for a better visualization.

Viana and Haftka (2010) proposed a multi-point algorithm based on an approximation of PI as infill criterion. Chaudhuri and Haftka (2012) proposed the algorithm EGO-AT by exploring the concept of targets for selecting multiple points (PI), discussed by Jones (2001). With EGO-AT (EGO with adaptive target) it is possible to adapt the targets for each optimization cycle based on the success of meeting the target in the previous cycle and generate multiple infill points.

As discussed in Haftka et al (2016), the seek of theoretical convergence proofs and rates that quantify the benefits of parallel computation constitutes important recent developments in the field. In this sense, several publications that investigate the theoretical bounds and rates of convergence and algorithm properties can be listed and it is remarkable the work of Desautels et al (2012) in machine learning area. Although the relevant theoretical developments, these studies and proposed algorithms did not provide yet superior performance against the other heuristic approaches that do not have proof of convergence.

Finally, according to Haftka et al (2016) the field of parallel surrogate-assisted global optimization with expensive models is a relatively new research field that is not mature yet and it is not possible to conclude with respect to the comparative efficiency of different approaches or algorithms. Further research is needed in order to take full advantage of additional improvements provided by parallelized surrogate based global optimization approaches.

In a different way, Viana et al (2013) proposed the MSEGO (multiple surrogates EGO). In this case, they used multiple surrogates simultaneously at each cycle of EGO algorithm. With at least one kriging model available, they imported the uncertainty estimate for this model to the other non-kriging models in the set. By means of different uncertainty estimates, they generate different instances of expected improvement functions to be maximized and to provide multiple parallel infill points in each EGO cycle.

Since we have an arbitrary set of M distinct metamodels, that are relatively fast to generate (as compared with the true simulation model $y(\mathbf{x})$), then it is possible to create an arbitrary number of N_p *partial* LS ensembles $\hat{y}^k_{ens}(\mathbf{x})$, by generating permutations of $\bar{M} < M$ metamodels. Therefore, by means of the N_p partial LS ensembles there are N_p respective expected improvement functions $E^k[I(\mathbf{x})]$ available to generate up to N_p infill points per cycle.

In this case, differently from Viana et al (2013), it is not required to have at least one kriging model in the set to generate the uncertainty estimates, since in LSEGO $\hat{s}^2_k(\mathbf{x})$ are generated directly from the least squares definition for the partial ensembles. Based on our performed numerical performed we observed a good performance of LSEGO for the purpose of multiple infill points per cycle.

The LSEGO algorithm will be summarized in the Section 7.1 and we will illustrate the behavior of LSEGO for one and two variable problems in Section 7.2.

7.1 LSEGO Algorithm with Parallel Infill Points

The LSEGO algorithm with parallel infill points can be summarized in the following steps:

i. Define a set χ of N sampling points and start the optimization cycles ($j \leftarrow 1$);

ii. Evaluate the true response $y(\mathbf{x})$ at all data sites in χ, at the current cycle j, and set

$$y_{min} \leftarrow \min(y_1, \cdots, y_N);$$

iii. Generate the M metamodels $\hat{y}_i(\mathbf{x})$, the N_p partial LS ensembles $\hat{y}^k_{ens}(\mathbf{x})$ and the respective expected improvement functions $E^k[I(\mathbf{x})]$, with all data available at the current cycle j;

iv. Find the set of next distinct $N^*_p \leq N_p$ infill points $\chi_{infill} \leftarrow \left[\mathbf{x}_{N+1}, \cdots, \mathbf{x}_{N+1+N^*_p}\right]$ as the respective maximizers of $E^k[I(\mathbf{x})]$; evaluate the true function $y\left(\mathbf{x}\right)$ at the N^*_p infill points and add them to the sampling space: $\chi \leftarrow \chi \cup \chi_{infill}$;

v. If the stopping criteria is not met, set

$$y_{min} \leftarrow \min(y_1, \cdots, y_{N+N^*_p}),$$

set ($N \leftarrow N + N^*_p$), update cycle counter ($j \leftarrow j + 1$) and return to step iii. Otherwise, finish the LSEGO algorithm.

As stated before, we extended the LSEGO algorithm in order to handle constrained optimization. In this case, the five steps outlined before are nearly the same for the unconstrained case. The only difference is that the metamodel for the constraints, i.e., $\hat{g}_i(\mathbf{x})$, should be created as well and the maximization should be taken by considering the constrained expected improvement defined by means of Eq. (27). We followed the directions provided with the implementations of EGO in Forrester et al (2008) to implement and extend our LSEGO algorithm.

In the next section we will first illustrate the behavior of LSEGO in the unconstrained optimization of one and two dimensional functions. In sequence, in Section 7.3 we will summarize the results of Ferreira and Serpa (SMO-15-0339). Further details can be found in the complete manuscript in Appendix G.

After that, the numerical experiments for constrained examples will be presented in Section 8.

7.2 Illustrations: One and Two Variables Examples

Fig. 23 illustrates the evolution of LSEGO for a one variable function with one infill point per optimization cycle. At each optimization cycle it is presented the true function $y(x)$ versus the approximation by LS-a ensemble (left plot) and the expected improvement function $E[I(x)]$ (right plot), calculated by means of Eq. (25), with $\hat{s}^2(\mathbf{x})$ as defined in Eq. (21).

At cycle 01 we have a very poor approximation with correlation coefficient $R^2 = 0.258$ and normalized root mean squared error $NRMSE = 31.9\%$. The $E[I(x)]$ presents a clearly defined peak close to the true minimum ($x_{opt}^{exact} = 5.624$). Note for this example in Fig. 23, at the first six optimization cycles, the maximum of expected improvement function works in the "exploitation mode" and prioritizes to add infill points around the optimum.

At this stage (cycle 06) the optimum found $x_{opt} = 5.600$ is quite close to the exact value (0.43% error), with a very good quality approximation for the metamodel: $R^2 = 0.975$ and $NRMSE = 4.2\%$. After cycle 06, the LSEGO algorithm automatically switches to the "exploration mode" and the infill points are selected in order to improve the quality of the approximation in the whole design space, instead of improving the minimum value found. The LSEGO algorithm was stopped at cycle 12 with $x_{opt} = 5.600$, $R^2 = 0.999$ and $NRMSE = 0.5\%$.

This behavior of LSEGO in one dimension was observed for other functions as well, with different levels of nonlinearity and multimodality. Based on these preliminary results we can conclude that LSEGO performed quite well in terms of exploitation and exploration of the design space.

On the other hand, for higher dimension problems we noted a very slow convergence for the algorithm, associated to a high concentration of infill points around the global optimum, as observed in standard EGO algorithm as well. See for instance in Fig. 24 the behavior for LSEGO for Giunta-Watson function (see Appendix D, Eq. (41)) with two variables and one infill point per optimization cycle. Note in the detail at Fig. 25 that the exact minimum is accurately found only at cycle 37 and all the infill points are located at this neighborhood. As consequence, the quality of the approximation at cycle 37 is still poor outside the optimum vicinity (note the difference on the exact and approximate contours for the function).

As we discussed before, note that the expected improvement function $E[I(x)]$ has shown to be very multimodal in most of the optimization cycles investigated for LSEGO (see the $E[I(x)]$ curves in Fig. 23). At higher dimensions it is expected that this effect can be more pronounced.

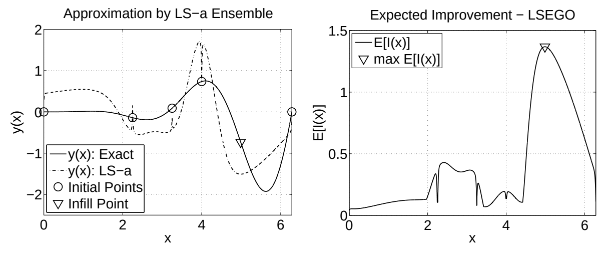
We will illustrate the behavior of LSEGO with multiple infill points with functions of two variables. Fig. 26 presents the same setup used in the case of Fig. 24 for Giunta-Watson function (2 variables), but now with $N_p = 8$ infill points per optimization cycle.

Note in Fig. 26 and details at Fig. 27 that, by allowing more infill points per cycle, LSEGO converges very quickly to the exact optimum at cycle 05 (as compared to cycle 37, for $N_p = 1$ in Fig. 24), with a reasonable correlation for the metamodel at this point ($R^2 = 0.803$ and $NRMSE = 8.1\%$). In addition, if we let LSEGO to continue with the exploration, the metamodel quality is continuously improved. The results at cycle 12 ($R^2 = 0.998$ and $NRMSE = 0.6\%$) in Fig. 27 can be explained by the more spread of infill points (exploration) not only on the vicinity of the optimum (exploitation), as in the case for one infill point per cycle.

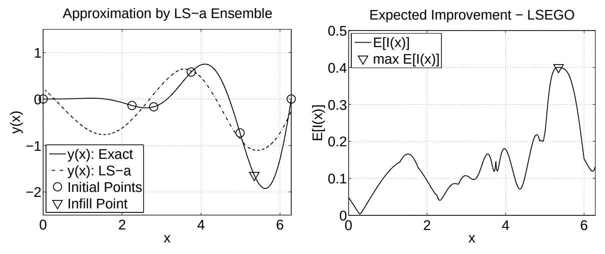
We observed the same performance of LSEGO for other functions as well. See for instance the evolution for Branin-Hoo function (ref. App. D, Eq. (39)) in Fig. 28 and details at Fig. 29. In this case, LSEGO has found the three optimum points within high accuracy at cycle 05 ($R^2 = 0.999$ and $NRMSE = 0.5\%$), see Fig. 29.

Based on these preliminary results with one and two dimension functions, we can infer that LSEGO has a good performance on driving EGO algorithm, with single and multiple infill points per cycle. As more infill points are added per cycle, faster is the convergence to the global optimum (exploitation) and also the quality improvement (predictability) of the metamodel in the whole design domain (exploration).

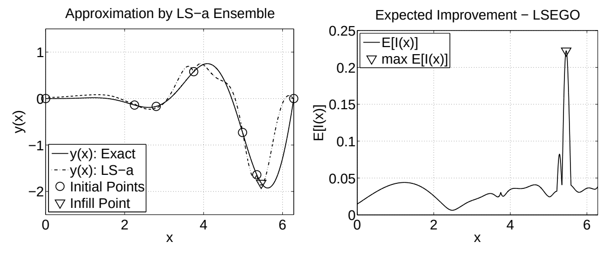
In the next section we will summarize the results of the numerical experiments performed with the objective to compare the performance of the proposed algorithm LSEGO versus the traditional EGO.



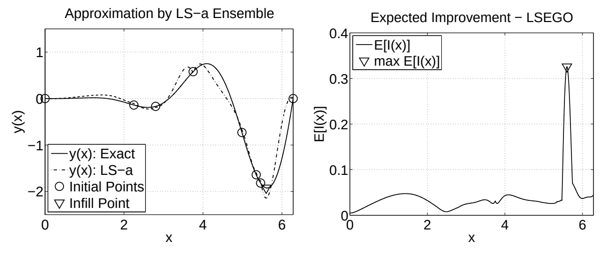
(a) Cycle 01



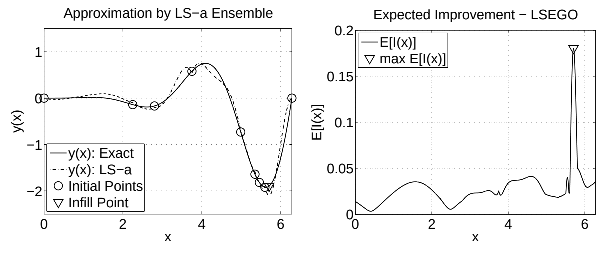
(b) Cycle 02



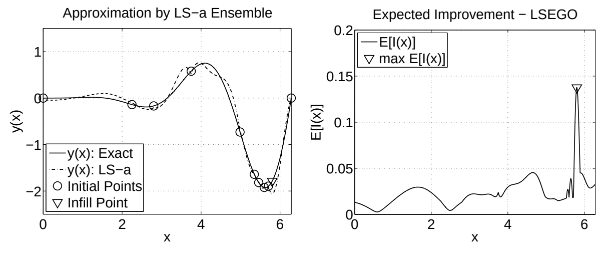
(c) Cycle 03



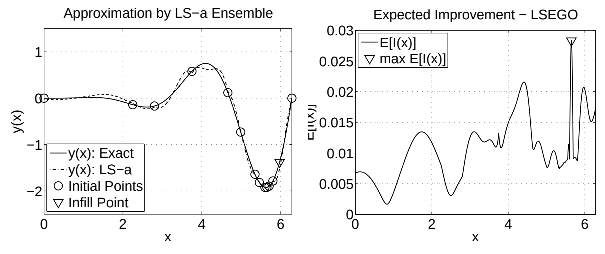
(d) Cycle 04



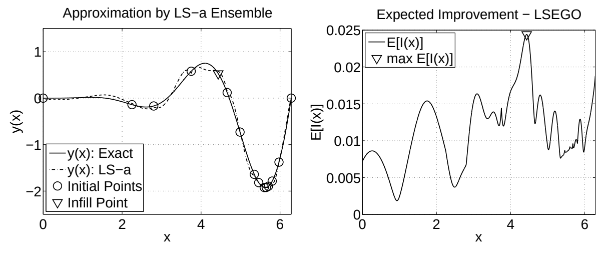
(e) Cycle 05



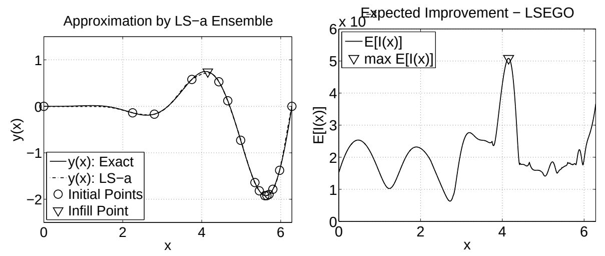
(f) Cycle 06



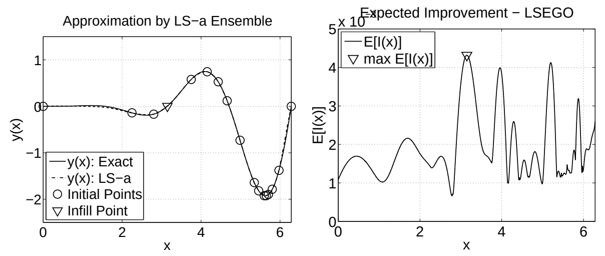
(g) Cycle 08



(h) Cycle 10

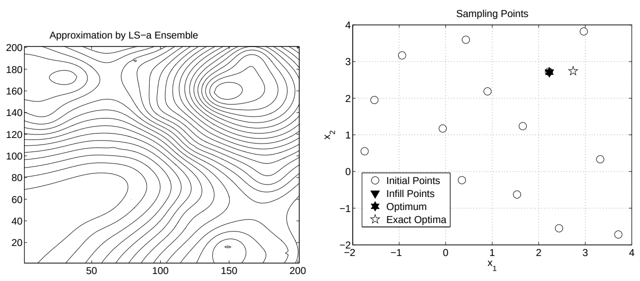


(i) Cycle 11

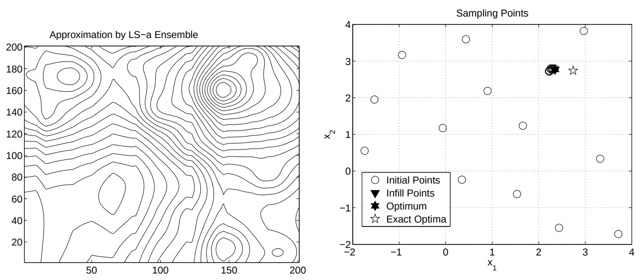


(j) Cycle 12

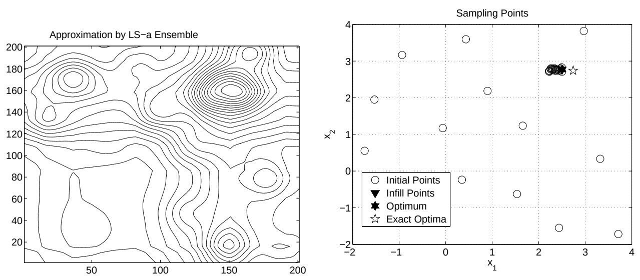
Figure 23: Evolution of approximation ($y(x)$ vs. $\hat{y}(x)$ and expected improvement $E[I(x)]$ for $y(x) = \frac{1}{700}(-2x+5x^2+7x^3)\sin(2x)$ with $N_p = 1$ infill point per optimization cycle with LSEGO algorithm. The initial sampling points are at $\chi = [0, 2.25, 2.8, 3.75, 2\pi]$ and the LS-a ensemble is used with four metamodels: PRS ($ID = 1$), KRG ($ID = 2$), RBNN ($ID = 3$) and SVR ($ID = 4$), see Tab. 7.



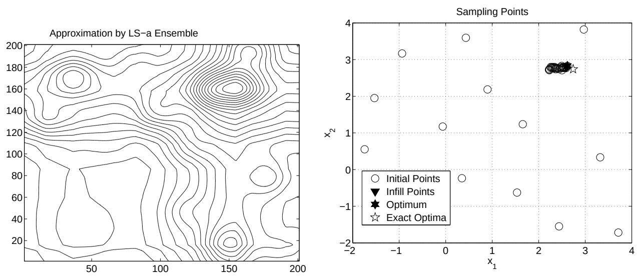
(a) Cycle 01



(b) Cycle 10

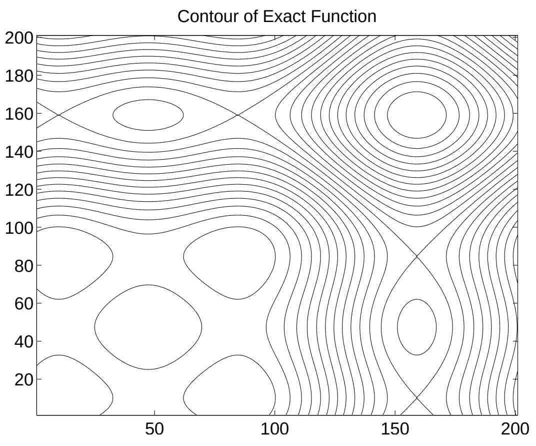


(c) Cycle 20

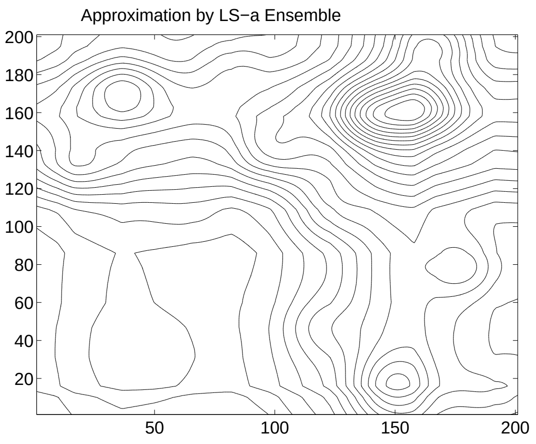


(d) Cycle 30

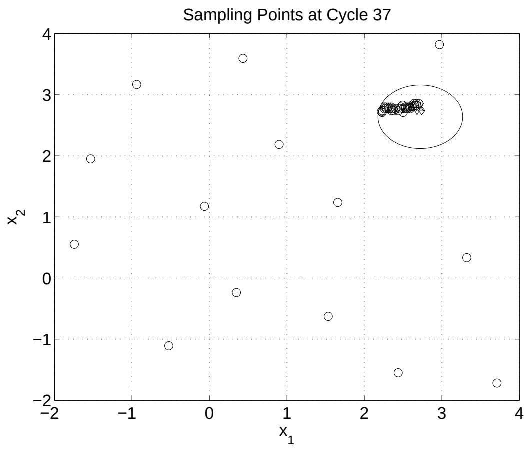
Figure 24: Evolution of approximation of Giunta-Watson function (2 variables) with LSEGO and $N_p = 1$ infill point per optimization cycle. The 15 initial sampling points were generated with Latin Hypercube Matlab function `lhsdesign`, and the LS-a ensemble is used with four metamodels: PRS ($ID = 1$), KRG ($ID = 2$), RBNN ($ID = 3$) and SVR ($ID = 4$), see Tab. 7.



(a) Exact Contour

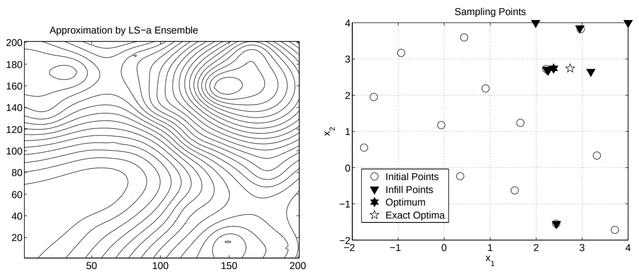


(b) Cycle 37: Approximation

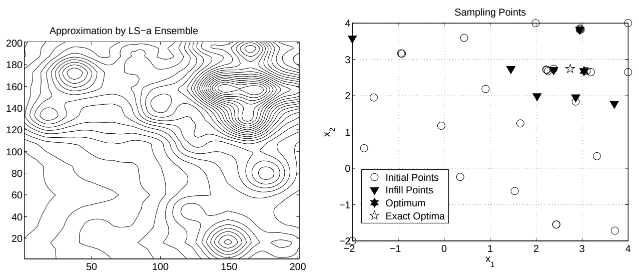


(c) Cycle 37: Sampling Points

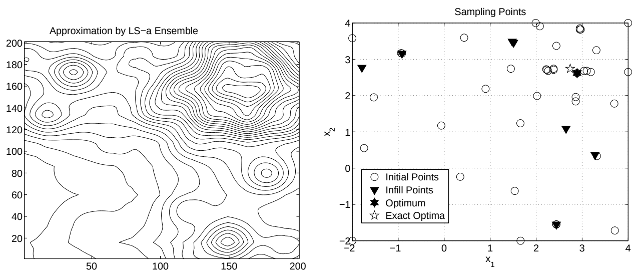
Figure 25: Detail of Fig. 25 at Cycle 37 for the evolution of approximation of Giunta-Watson function (2 variables) with LSEGO and $N_p = 1$ infill point per optimization cycle.



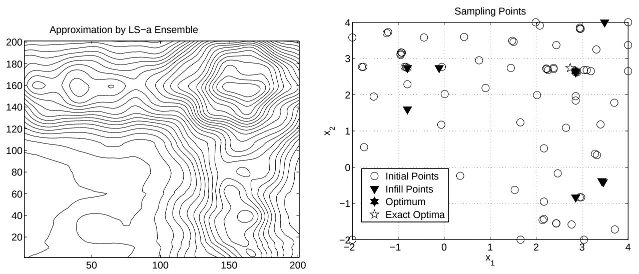
(a) Cycle 01



(b) Cycle 03

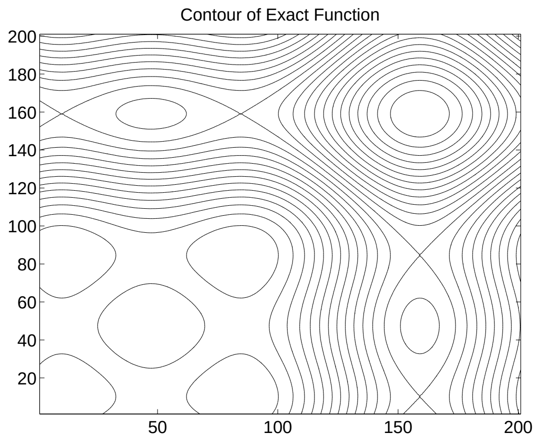


(c) Cycle 05

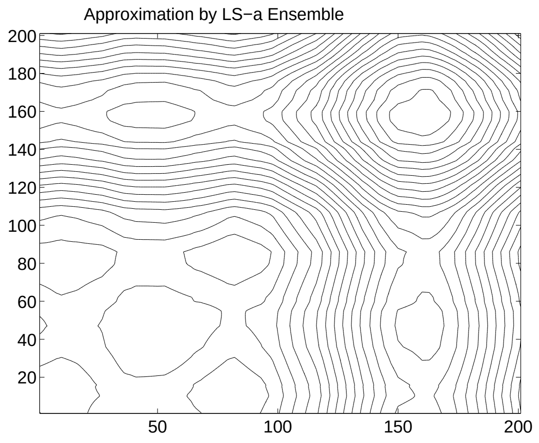


(d) Cycle 10

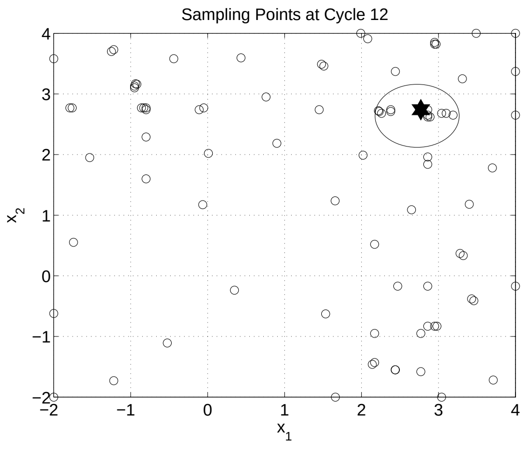
Figure 26: Evolution of approximation of Giunta-Watson function (2 variables) with LSEGO-8: $N_p = 8$ infill points per optimization cycle and the LS-a ensemble is used with four metamodels: PRS ($ID = 1$), KRG ($ID = 2$), RBNN ($ID = 3$) and SVR ($ID = 4$), see Tab. 7.



(a) Exact Contour

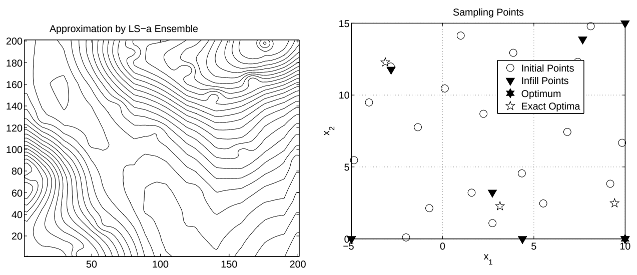


(b) Cycle 12: Approximation

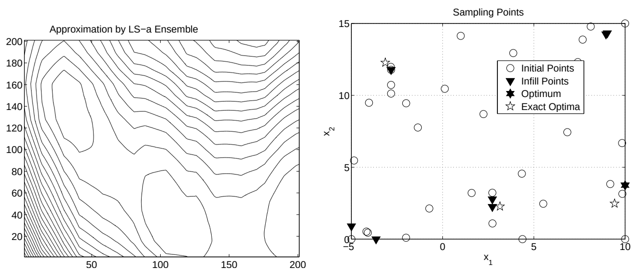


(c) Cycle 12: Sampling Points

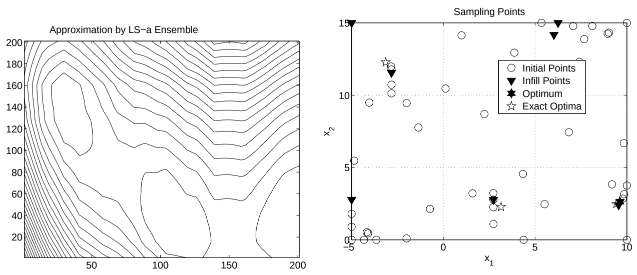
Figure 27: Detail of Fig. 26 at Cycle 12 for the evolution of approximation of Giunta-Watson function (2 variables) with LSEGO-8: $N_p = 8$ infill points per optimization cycle.



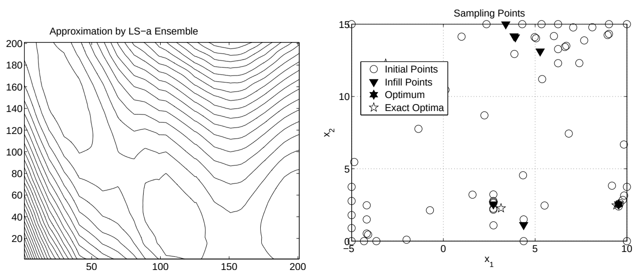
(a) Cycle 01



(b) Cycle 03

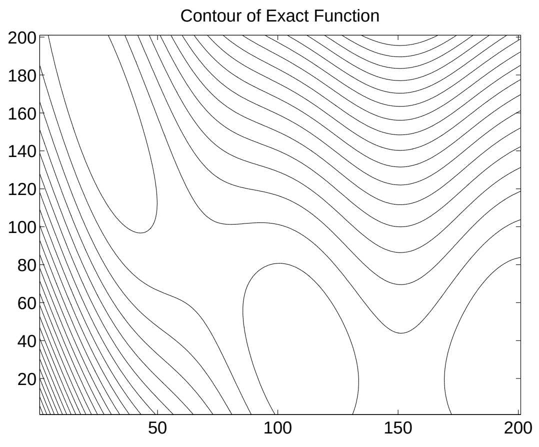


(c) Cycle 05

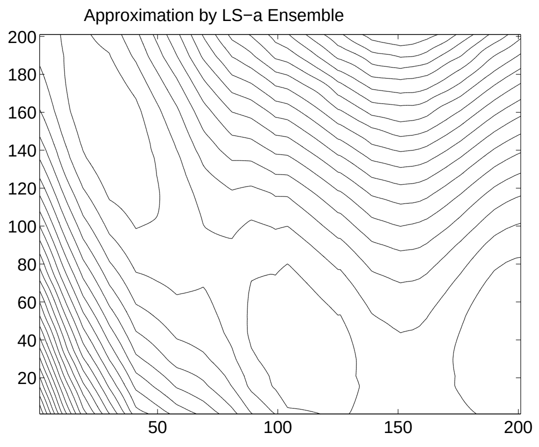


(d) Cycle 10

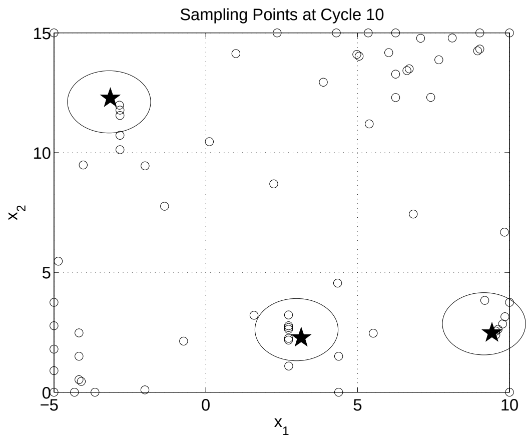
Figure 28: Evolution of approximation of Branin-Hoo function (2 variables) with LSEGO-8: $N_p = 8$ infill points per optimization cycle and the LS-a ensemble is used with four metamodels: PRS ($ID = 1$), KRG ($ID = 2$), RBNN ($ID = 3$) and SVR ($ID = 4$), see Tab. 7.



(a) Cycle 10: Exact Contour



(b) Cycle 10: Approximation



(c) Cycle 10: Sampling Points

Figure 29: Detail of Fig. 28 at Cycle 10 for the evolution in the approximation of Branin-Hoo function (2 variables) with LSEGO-8: $N_p = 8$ infill points per optimization cycle.

7.3 The LSEGO Approach: Results Summary

We presented LSEGO, an approach to drive efficient global optimization (EGO), based on LS (least squares) ensemble of metamodels. By means of LS ensemble of metamodels it is possible to estimate the uncertainty of the prediction by using any kind of metamodels (not only kriging) and provide an estimate for the expected improvement function. In this way, LSEGO arises as an alternative to find multiple infill points at each cycle of EGO and improve both convergence and prediction quality during the whole optimization process.

At first, we demonstrated the performance of the proposed LSEGO approach with one dimensional and two dimensional analytical functions (these results were presented and discussed in Section 7.2). The algorithm has been tested with increasing number of infill points per optimization cycle. As more infill points are added per cycle, faster is the convergence to the global optimum (exploitation) and also the quality improvement (predictability) of the metamodel in the whole design domain (exploration).

In a second test set, we compared the proposed LSEGO approach with the traditional EGO (with kriging and a single infill point per cycle). For this intent, we used well known analytical benchmark functions to test optimization algorithms, from two to six variables.

For the problems studied, the proposed LSEGO algorithm has shown to be able to find the global optimum with a much smaller number of optimization cycles required by the classical EGO approach. This accelerated convergence was specially observed as the number of variables increased, when the standard EGO can be quite slow to reach the global optimum. See these results in Fig. 30 for the functions Branin-Hoo, Hartman-3 and Hartman-6.

The results also showed that, by using multiple infill points per optimization cycle, driven by LSEGO, the confidence of metamodels prediction in the whole optimization process is improved. It was observed in the boxplots for all cases investigated a variability reduction with respect to the initial sampling space (initial DOE) as more infill points are added during the optimization cycles.

Fig. 31 presents boxplots of the optimization results for 100 experimental designs for LSEGO vs. EGO. Note that at the beginning of the optimization (cycle 1) the minimum solution found has a high dispersion as function of the initial DOE and this dispersion decreases rapidily for LSEGO as the optimization cycles evolve. This behavior is observed in EGO but at a much lower rate.

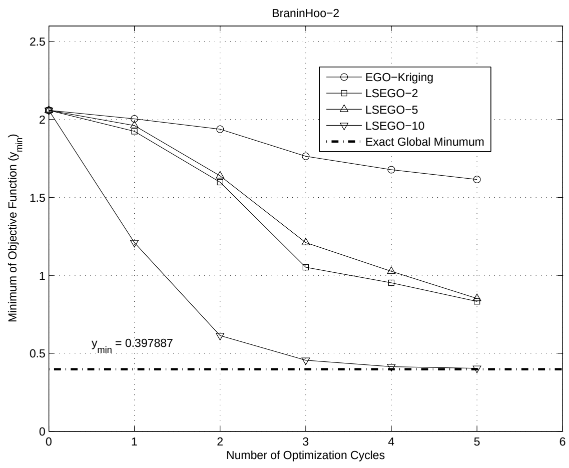
In the same direction of the results presented by Viana et al (2013), we found that by using LSEGO with multiples points per cycle, a significant reduction in the dispersion of the results (minimum of objective function) can be achieved as long as the optimization cycles evolve. This trend confirms that by using LSEGO with multiple points per cycle the variability and dependence of the optimization results on the initial DOE are reduced, what is not assured by using EGO with only one infill point per cycle.

We also compared LSEGO versus standard EGO in terms of the number of function evaluations, which translates directly to computational cost (i.e., number of simulations required). In the cases investigated, for two and six variables LSEGO performed better than EGO, but for three variables EGO performed better than LSEGO. Therefore, if there is no restriction on the number of parallel simulations, LSEGO accelerates significantly the convergence, but on the other hand, the number of total function evaluations is problem dependent (on both the dimension and nonlinearity) and the cost can be higher than required by EGO, for the same level of improvement in the objective function.

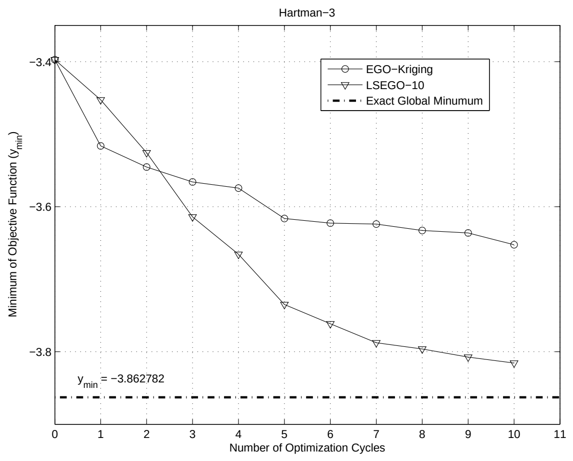
Recall that, as we discussed previously in Section 2.2.2, when parallel computation is an easy available resource, the potential penalty of parallel EGO approaches in terms of number of function evaluation should be neglected in favor of fast delivering optimization results.

Finally, the results achieved are in accordance with previous work published in the related research area. In this way, LSEGO approach has shown to be a feasible alternative to drive efficient global optimization by using multiple or ensemble of metamodels, not restricted to kriging approximation or single infill point per optimization cycles.

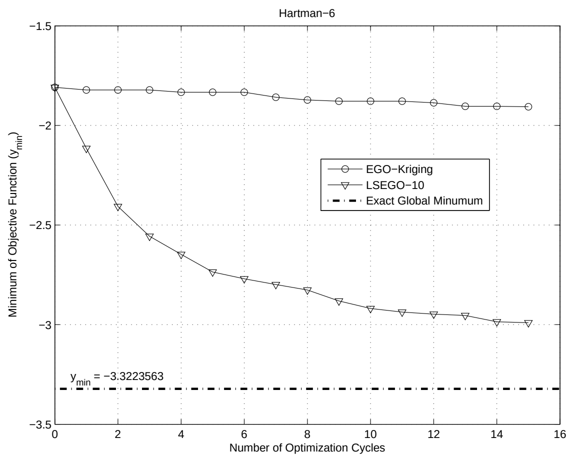
As discussed before, we extended the implementations for the standard LSEGO in order to handle constrained optimization (see Eq. (27)). In order to test the effectiveness of this extension to LSEGO approach in constrained optimization, some numerical experiments were performed with analytical benchmark functions and engineering problems. These new results will be described in detail in the Chapters 8 and 9.



(a) Branin-Hoo, $n_v = 2$

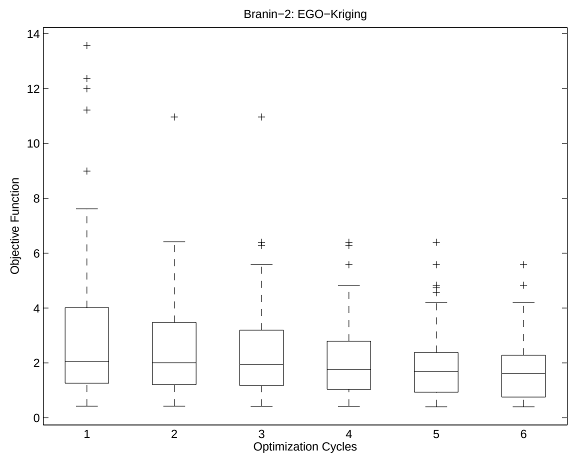


(b) Hartman-3, $n_v = 3$

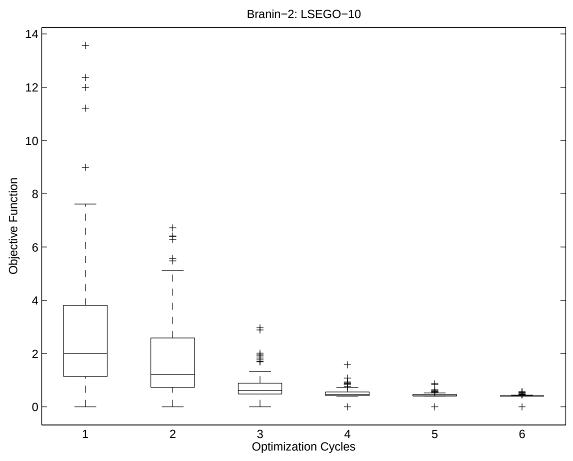


(c) Hartman-6, $n_v = 6$

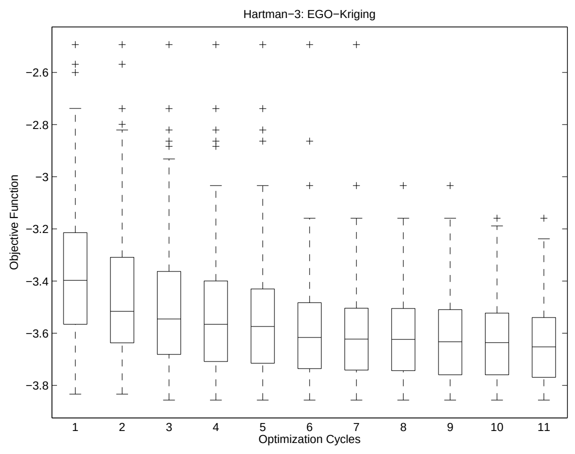
Figure 30: Comparison EGO-Kriging versus LSEGO variants (LSEGO-x, where "x" indicates the number of points per cycle) for the functions Branin-Hoo, Hartman-3 and Hartman-6. Median (over 100 different initial sampling, DOE) for the efficient global optimization results as function of the number of cycles. The convergence to the exact global minimum is accelerated by adding more points per cycle with LSEGO in all the cases studied.



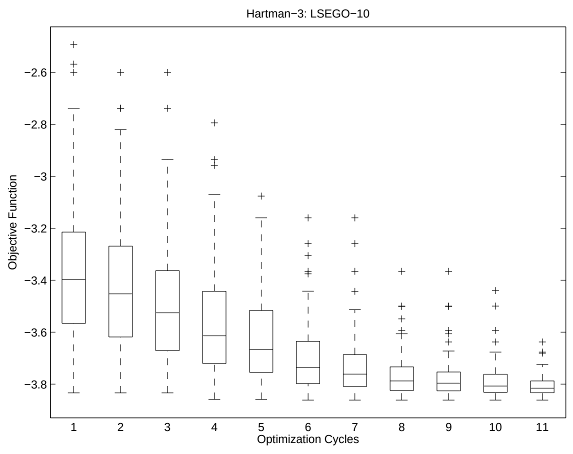
(a) Branin-Hoo, EGO



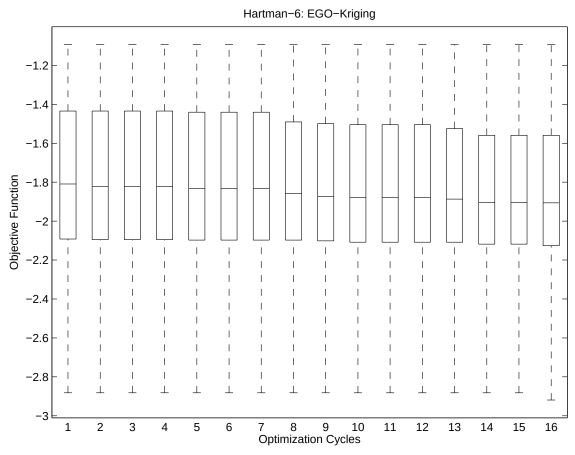
(b) Branin-Hoo, LSEGO-10



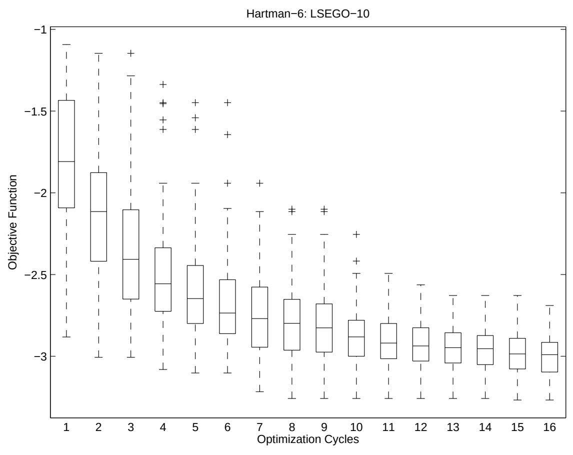
(c) Hartman-3, EGO



(d) Hartman-3, LSEGO-10



(e) Hartman-6, EGO



(f) Hartman-6, LSEGO-10

Figure 31: Comparison of the convergence of EGO-Kriging (left boxplots) vs. LSEGO with ten points per cylce, LSEGO-10 (right boxplots), over 100 different inital DOE in each case for the functions Branin-Hoo, Hartman-3 and Hartman-6. The variability of the results is higher as the number of variables increases and the convergence to the optimum is very slow with one infill point and EGO-Kriging. In all cases, the addition of multiple infill points per optimization cycle with LSEGO-10 accelerates the convergence and also reduces significantly the dispersion of the results as the optimization cycles evolve.

8 Applications to Constrained Optimization Problems

"It doesn't matter how beautiful your theory is.
It doesn't matter how smart you are.
If it doesn't agree with experiment, it's wrong."
Richard P. Feynman, 1918-1988.

The objective of this section is to present the implementation of LSEGO for handling constrained optimization problems. As discussed in Section 1.3, this is part of the **Phase III** of the research.

The first numerical test set was based on analytical benchmark functions. This kind of functions are widely used to validate both metamodeling and optimization methods. We tested before these functions in the previous phases of the work and we will extend the analysis here in the context of constrained optimization.

The objective of the second test set is to apply LSEGO in some analytical problems well known and published in the engineering literature. These problems have number of variables, nonlinear characteristics and number of constraints typical of practical engineering problems.

With both analytical benchmarks and engineering functions, we expect to understand the behavior and to validate the LSEGO algorithm with functions that are easy to generate the sampling points we need with virtually no computer cost. In addition, for these analytical functions we can know in advance the local or global optimum points based on previous results published in the optimization literature. Furthermore, we also can easily generate "reference optimum" points, by using any available optimization procedure or software as Matlab Optimization Toolbox, for instance.

In the sequence, let us remember that the main reason for using a metamodel based optimization process is when the true analytical functions are not known in advance, like the ones defined by computer codes of engineering applications (e.g., FE and CFD simulation models), as we introduced in Section 2.2.1 and detailed in Chapter 3.

In this sense, the objective of the last numerical experiment is to exemplify the application of LSEGO in the constrained optimization of an industry scale engineering problem, based on a finite element computer simulation model.

The present section will be divided as follows. In Section 8.1 it is presented the methodology used for computer implementation of LSEGO algorithm. In Section 8.2 the chosen test problems are described and the details of the design of experiments are presented in Section 8.3. The results of the numerical experiments performed are presented and discussed in Chapter 9.

8.1 Computer Implementation

We used the Matlab based SURROGATES Toolbox v2.0 (ref. Viana (2009)) as platform for implementation and tests. See Appendix A for details.

In Ferreira and Serpa (2016), we implemented routines for LS ensemble of metamodels, as described in Section 5. In Ferreira and Serpa (SMO-15-0339), refer to Appendix G, we extended the implementations to include the standard EGO and LSEGO algorithms as well, as described respectively in sections 6.2 and 7.1.

As presented and discussed in Section 7.1, in the present work we extended implementation of our prior LSEGO algorithm to handle constrained optimizations by using Eq. (27).

The numerical implementation has been performed with Matlab v2009 on a computer Intel(R) Core(TM) i7-3610QM, CPU 2.30GHz, 8Gb RAM, 64bits, and operational system Windows 7.

8.1.1 Ensembles of Metamodels

The ensembles of metamodels were created with the augmented least squares approach LS-a (Section 5), with $\eta_{aug} \approx 33\%$ and nine distinct models of type PRS, KRG, RBNN and SVR, by considering the setup presented in Tab. 7. Refer to SURROGATES Toolbox manual (ref. Viana (2009)) for details on the equations and tuning parameters for each of these metamodeling methods.

In order to create the partial ensembles for LSEGO, we used ten permutations of the nine models displayed on Tab. 2, as follows. The first (full) ensemble used all the nine metamodels. For the second partial ensemble we removed the model with $ID = 9$ from the full ensemble. For the third one, the model with $ID = 8$ was removed from the full ensemble and we continued this way up to get ten ensembles, i.e., one full LS ensemble ($M = 9$) and nine partial LS ensembles ($\bar{M} = 8$).

Table 2: Basic metamodels setup for creating the ensembles.

ID	Type	Details
1	PRS	Full quadratic model
2	KRG	Quadratic regression, exponential correlation, $\theta_0 = 10$ and $10^{-2} \leq \theta_i \leq 200$
3	RBNN	$Goal = (0.05\bar{y})^2$, $Spread = 2/5$ and $MN = N$
4	SVR	$C = 100max(\|\bar{y} + 3\sigma_y\|, \|\bar{y} - 3\sigma_y\|)$ and $\epsilon = \sigma_y/\sqrt{N}$
5	PRS	Linear model
6	KRG	Linear regression, Gaussian correlation, $\theta_0 = 10$ and $10^{-2} \leq \theta_i \leq 200$
7	RBNN	$Goal = (0.05\bar{y})^2$, $Spread = 1/3$ and $MN = N/2$
8	SVR	$C = 100max(\|\bar{y} + 3\sigma_y\|, \|\bar{y} - 3\sigma_y\|)$, $\epsilon = \sigma_y/\sqrt{N}$, $KernelOptions = 1/2$ and $Loss = Quadratic$
9	KRG	Constant regression, Gaussian correlation, $\theta_0 = 10$ and $10^{-2} \leq \theta_i \leq 200$

Obs.1: KRG: kriging; PRS: polynomial response surface; RBNN: radial basis neural network; SVR: support vector regression.
Obs.2: All other parameters not mentioned are kept with default values.
Obs.3: $\bar{y}$, σ_y and N are respectively: mean and standard deviation of y and number of sampling points.
Obs.4: No attempt has been made in order to fine tuning the surrogates shape parameters.

8.1.2 Maximization of the Expected Improvement Functions

In case of standard EGO, to miximize the expected improvement function at each optimization cycle, we used the Matlab Genetic Algorithm `ga`, with `InitialPopulation` set as $10n_v$ individuals, chosen by using the function `lhsdesign`, optimized with `maxmin` criterion set to 1000 iterations. The `PopulationSize` and `Generations` options were set to 100, with the other parameters as default.

In case of LSEGO with multiple infill points, there are many expected improvement functions to maximize per cycle. The use of a genetic algorithm can be quite time consuming in this case. Based on preliminary numerical experiments, we found a good balance in accuracy and computation time by using the Matlab Pattern Search algorithm `patternsearch`, with $10n_v$ initial points (`X0`), chosen by using the function `lhsdesign`, optimized with `maxmin` criterion set to 1000 iterations.

As discussed in Jones (2001), the EGO-type algorithms tends to generate infill points quite close to each other in several cycles. Sampling points too close can degenerate the approximation of many metamodels, in special KRG and RBF.

In addition, infill points too close have low contribution to the exploration of the design space, they can be a waste of resources and, in addition, they lead to a slower convergence for the optimization. We verified this fact in several preliminary numerical experiments (ref. for instance the illustration example of Fig. 25).

In this case, in order to avoid approximation issues during LSEGO cycles, we generated exceeding infill points per cycle and selected *distinct* N_p with highest expected improvement value.

In our tests, we found a good balance by generating $2N_p$ candidate infill points and then we used a *clustering* procedure to remove points too close, or even equal to each other. After the clustering selection, the N_p distinct points are added to the sampling space for the next optimization cycle.

In our first tests we used the Matlab function `unique` to remove equal points from the sampling space, but this procedure has shown to be not effective.

Then we implemented a clustering selection procedure by using the Matlab function `cluster` with the `distance` criterion. The `cutoff` was based on the maximum distance (d_{max}) among points/clusters in the whole sampling space by using the Matlab `linkage` function. In preliminary numerical tests we found that `cutoff` around to 10% of d_{max} is effective to remove too close points.

8.2 Test Problems

8.2.1 Analytical Benchmark Functions

We used three well known analytical functions with different number of variables (n_v) for testing the optimization algorithms: Branin-Hoo ($n_v = 2$), Hartman-3 ($n_v = 3$) and Hartman-6 ($n_v = 6$). See Appendix D for the respective equations.

For the analytical benchmark functions, we generated the constraints by following the approach used in Forrester et al (2008) to test the constrained expected improvement formulation.

That is, for Branin-Hoo function, let $x_1^* = 3\pi$ and $x_2^* = 2.475$ be the coordinates of one of the three local optima, then we write the normalized constraint as follows

$$g(x_1, x_2) = \frac{x_1 x_2}{x_1^* x_2^*} - 1 \geq 0. \tag{28}$$

In this way, by using this hyperbolic function, at least one local optimum is forced to lie exactly at the constraint boundary.

The same idea was used for Hartman-3 and Hartman-6 functions, with their respective global optima, listed in Appendix D.

8.2.2 Analytical Engineering Problems

Four analytical constrained optimization problems, available in the mechanical engineering technical literature, were used to assess the performance of the proposed LSEGO algorithm.

Three-Bar Truss This is a test problem for constrained optimization, available in the book Rao (2009). The objective functions to be minimized are the total weight $f_1(\mathbf{x})$ and the tip displacement $f_2(\mathbf{x})$ of a simple three bar truss. The design variables are two distinct cross section areas (x_1 and x_2) and the constraints are the tensile stresses at the three bars. i.e., $\sigma_1(\mathbf{x})$, $\sigma_2(\mathbf{x})$ and $\sigma_3(\mathbf{x})$. The respective equations are:

$$\begin{aligned}
f_1(\mathbf{x}) &= 2\sqrt{2}x_1 + x_2 \\
f_2(\mathbf{x}) &= \frac{PH}{E}\frac{1}{x_1+\sqrt{2}x_2} \\
\sigma_1(\mathbf{x}) &= P\frac{x_2+\sqrt{2}x_1}{\sqrt{2}x_1^2+2x_1x_2} - \sigma^{(u)} \quad \leq 0 \\
\sigma_2(\mathbf{x}) &= P\frac{1}{x_1+\sqrt{2}x_2} - \sigma^{(u)} \quad \leq 0 \\
\sigma_3(\mathbf{x}) &= -P\frac{x_2}{\sqrt{2}x_1^2+2x_1x_2} - \sigma^{(l)} \quad \leq 0
\end{aligned} \quad , \tag{29}$$

$$x_i^{(l)} \leq x_i \leq x_i^{(u)} \quad i = 1,2$$

where P is the applied force, H is the the main dimension of the truss and E is the Young's modulus. The numeric data are: $\sigma^{(u)} = 20$, $\sigma^{(u)} = -15$, $x_i^{(l)} = 0.1$, $x_i^{(u)} = 5.0$, $P = 20$, $H = 1$ and $E = 1$.

For f_1, the reported minimum is

$$\mathbf{x}^* = (0.78706, 0.40735)$$

with $f_1^* = 2.6335$ and for f_2, the reported minimum is

$$\mathbf{x}^* = (5, 5)$$

with $f_2^* = 1.6569$.

Cantilever Beam These functions are due to Nowacki (1980) and they are used as test set for metamodeling and optimization in Forrester et al (2008). The problem refers to a cantilever beam of constant length $l = 1.5$ m, with an end tip load $F = 5$ kN and the variables $\mathbf{x} = (b, h)$ are the retangular cross-section dimensions: the width ($5 \leq b \leq 50$) [mm] and the height

$(20 \leq h \leq 250)$ [mm]. The objective is to minimize the cross section area $y(\mathbf{x}) = bh$, subjected to the constraints for tip displacement $\delta(\mathbf{x})$ and bending stress $\sigma(\mathbf{x})$, i.e.,

$$\begin{aligned} \delta(\mathbf{x}) &= \frac{Fl^3}{3EI_Y} - \delta_{max} \quad \leq 0 \\ \sigma(\mathbf{x}) &= \frac{6Fl}{bh^2} - \sigma_Y \quad \leq 0 \end{aligned} \quad , \tag{30}$$

where $I_Y = \dfrac{bh^3}{12}$ is the section moment of inertia and $E = 216.62$GPa, $G = 86.65$GPa and $\nu = 0.27$ are the Young modulus, shear modulus and Poisson coefficient respectively for the beam material. $\sigma_Y = 240$ MPa is the yielding stress limit and $\delta_{max} = 5$ mm is the maximum allowed tip deflection. There is no reported optimum for this problem, then we used Matlab optimization function `fmincon` and found

$$\mathbf{x}^* = (5, 231.9)$$

with $y^* = 1159.3$, that will we considered as the "reference optimum" solution.

Helical Spring This is also an example problem available in Rao (2009). The design variables are the wire diameter d, the coil diameter D and the number of turns N. The objective is to minimize the spring weight $y(\mathbf{x})$, by limiting the deflection $\delta(\mathbf{x})$, shear stress $\tau(\mathbf{x})$ and vibration natural frequency $F_n(\mathbf{x})$. In summary, the equations are:

$$\begin{aligned} y(\mathbf{x}) &= \frac{\pi d^2}{4}\pi D N \rho \\ \delta(\mathbf{x}) &= \frac{8FD^3N}{d^4G} - 0.1 \leq 0 \\ \tau(\mathbf{x}) &= K_s\frac{8FD}{\pi d^3} - 10^4 \leq 0 \\ F_n(\mathbf{x}) &= \frac{\sqrt{Gg}d}{2\sqrt{2\rho}\pi}\frac{d}{D^2N} - 10^2 \geq 0 \\ & d,\ D,\ N > 0 \end{aligned} \quad , \tag{31}$$

where $F = 1250$lb, $\rho = 0.3$lb/in^3, $K_s = 1.05$, $G = 12 \times 10^6$psi and $g = 386.22$in/s^2.

There is no reported optimum for this problem then, with the same approach applied in the *Cantilever Beam*, we used Matlab optimization function `fmincon` and found

$$\mathbf{x}^* = (1, 1.5, 3)$$

with $y^* = 3.331$, that will we considered as the “reference optimum” solution.

Pressure Vessel This problem is acknowledged to Kannan and Kramer (1994). A cylindrical pressure vessel has its both ends capped by hemispherical heads. There are four geometrical design variables x_i (i.e., two thicknesses, length and radius) and four constraints $g_j(\mathbf{x})$. The objective is to minimize the manufacturing cost, $y(\mathbf{x})$, due to material and processes. The problem can be stated as:

$$\begin{aligned}
y(\mathbf{x}) &= 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 \\
&\qquad + 3.1661x_1^2x_4 + 19.84x_1^2x_3 \\
g_1(\mathbf{x}) &= -x_1 + 0.0193x_3 \leq 0 \\
g_2(\mathbf{x}) &= -x_2 + 0.00954x_3 \leq 0 \\
g_3(\mathbf{x}) &= -\pi x_3^2x_4 - \frac{4\pi}{3}x_3^3 + 1.296 \times 10^6 \leq 0 \\
g_4(\mathbf{x}) &= x_4 - 240 \leq 0 \\
& 0.0625 \leq x_{1,2} \leq 6.1875 \\
& 10 \leq x_{3,4} \leq 200
\end{aligned} \quad . \tag{32}$$

Kannan and Kramer (1994) reported the optimum for this problem as

$$\mathbf{x}^* = (1.125, 0.625, 58.291, 43.690),$$

with $y^* = 7198.0428$.

It is important to note that this is a highly multimodal problem, and several authors published different local optimum solutions in the neighborhood of the one reported by Kannan and Kramer (1994), in the range $(6000 < y^* < 9000)$.

In addition, most of these recent solutions for this problem were based on evolutionary optimization (i.e., genetic algorithms, etc.), by using true functions instead of metamodels, and the authors reported 80×10^3 to 150×10^3 function evaluations to reach the local optimum. See a collection of examples in Coello (2000), Lemonge et al (2010) and Yang et al (2013a).

We will consider

$$\mathbf{x}^* = (0.8125, 0.4375, 42.0984, 176.6366),$$

with $y^* = 6059.7143$, reported by Yang et al (2013a), as the "reference optimum" solution for comparison.

8.2.3 Engineering Application: Car Impact

The objective of this numerical experiment is to exemplify the application of LSEGO in the constrained optimization of an industry scale engineering problem, based on finite element computer simulation model.

We selected one example regarding the optimization of collapsible impact energy absorbers. The development of efficient impact absorbers is crucial in several engineering applications as for example crashworthiness design of vehicles (cars, lifts, aircraft, ships, etc.), crash barrier design, safety of nuclear structures, collision damage to road and bridges, offshore installations and many others. Refer to Alghamdi (2001) for a comprehensive review on this subject.

The computer simulation of collapsible impact energy structures involves several complex phenomena such as large displacement/rotations, large strains and nonlinear material behavior of components under contact and impact (transient loading).

There is a lot of research studies and engineering applications on this subject. See for instance some recent publications: Elmarakbi et al (2013), Zhang et al (2014), Tanlak and Sonmez (2014) and Zhang et al (2015).

Depending on the level of detail desired, the finite element model can take several minutes for a single run or, even hours or days in high-end parallel computer clusters, as we mentioned in Section 1.1. In such context, the metamodel approach is often the only feasible way to perform design optimization.

The chosen model is part of the tutorial examples available in the HyperWorks Student Edition, version v13.0 [9]. The models was built and executed with RADIOSS finite element

[9]HyperWorks is developed and distributed by Altair Engineering, Inc., see `www.altair.com`. HyperWorks

solver, that is largely used in industry for engineering applications, specially involving crashworthiness design.

The model is a simplified version for the full Car Frontal Pole Impact test, see Fig. 32. Car crash tests are important for both occupant and pedestrian safety and are part of regulations in many countries nowadays. For more information and details see the main regulation institutes, i.e., NCAP[10] and NHTSA[11].

The vehicle front end is main structure to sustain efficiently the impact energy during a frontal pole impact, for example. This kind of automotive structure is generally made by thin sheet metal parts, in form of tubes, assembled together with spot or seam welds. See Fig. 33 for details.

We defined four continuous thickness design variables (t_1, t_2, t_3 and t_4), in the range [1.0 to 3.0] mm, for optimization, with baseline values

$$\mathbf{x} = (2.5, 1.55, 1.55, 1.55).$$

In this case, the rigid wall is fixed and massless. The vehicle moves in the direction of the rigid wall with a initial velocity of 15.6m/s.

The objective is to maximize the specific energy absorbed (SEA) due to plastic deformation, by constraining the peak impact force to a controlled safety level. This approach is commonly used in crashworthiness design optimization, see for instance Zhang et al (2014).

In this example, we normalized specific energy absorbed by the baseline value $SEA_{baseline}$, then the objective function can be defined as

$$y(\mathbf{x}) = \frac{SEA(\mathbf{x})}{SEA_{baseline}}.$$

In the same way, the constraint function, i.e., the normalized maximum/peak impact force $F_{max}(\mathbf{x})$, with respect to the baseline value $F_{baseline}$, can be defined as

$$g(\mathbf{x}) = 1 - \frac{F_{baseline}}{F_{max}(\mathbf{x})} \geq 0.$$

This model was based on the tutorial example (HS-3550), available in HyperWorks. In

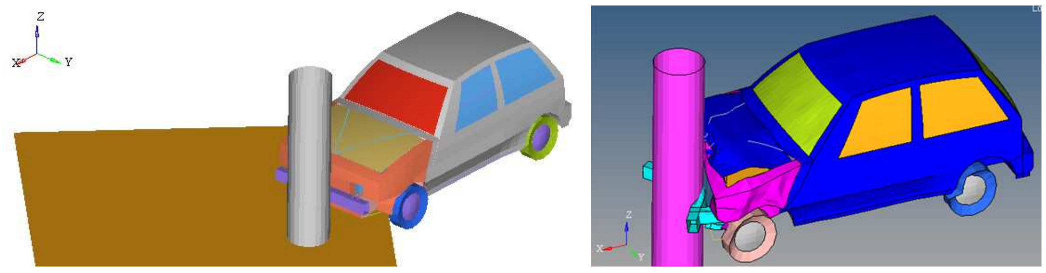
Student Edition was used in this work under authorization by Altair Engineering, Inc.

[10]`www.globalncap.org`

[11]`www.nhtsa.gov`

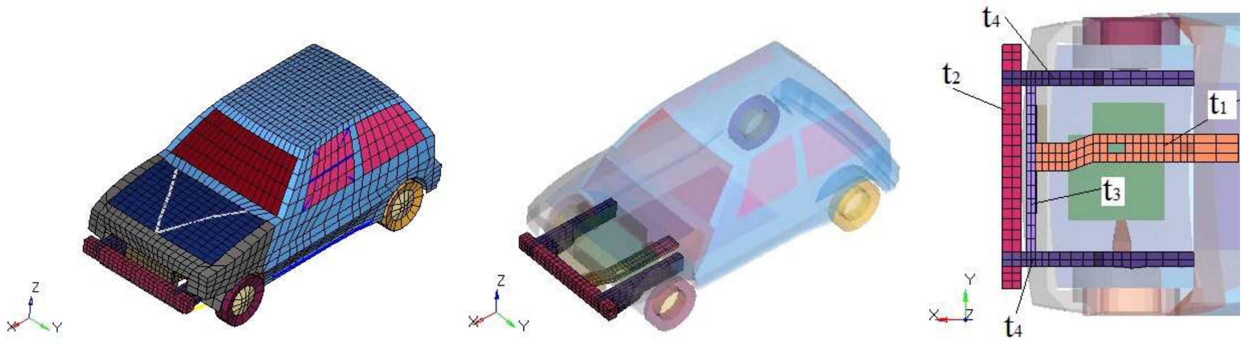
the computer system used in this work, each model takes about 10min for simulating 50ms of the impact event.

For additional model details, see the online HyperWorks documentation.



(a) Frontal Pole Impact Model (b) Deformed Shape

Figure 32: Car frontal pole impact model. Demo RADIOSS model (HS-3550) available in HyperWorks Student Edition, v13. Courtesy of Altair Eng., Inc.



(a) FE Mesh (b) Front End Structure (c) Design Variables

Figure 33: Car frontal pole impact model. Details of the finite element (FE) mesh, front end structure and design variables (thickness, t_1, t_2, t_3 and t_4) selected for optimization.

8.3 Design of Experiments (DOE)

The initial DOE with N points ($N = N_{tr} + N_{add}$) were created using the Latin Hypercube Matlab function `lhsdesign`, optimized with `maxmin` criterion set to 1000 iterations. At each cycle of LSEGO-10 (i.e., ten infill points per cycle), the augmenting points N_{add} were chosen from the full sampling set (N) to generate the LS-a ensemble with constant rate $\eta_{aug} \approx 33\%$.

After some preliminary numerical tests, we observed that it is important to assure for each cycle i that the current optimum point, i.e., $(\mathbf{x}^i_{min}, y^i_{min})$ is always in the training set N^i_{tr}. Otherwise, the expected improvement function might wrongly generate an optimum point y^{i+1}_{min}, with higher value of y^i_{min} for the next cycle.

In order prevent this, at each cycle i, we sort the N^i available sampling points in increasing order of $y(\mathbf{x})$ and put the current optimum at the top of the row. Then, the augmenting points N_{add} for the validation set are selected from the tail of the row, i.e., the ones with highest values of $y(\mathbf{x})$ and, in this way, the current optimum is always on the training set N^i_{tr}.

For the analytical benchmark functions, i.e., Branin-Hoo, Hartman-3 and Hartman-6, we used the same number of initial points N in the DOE as applied in Jones et al (1998), that is, $N = 21$, $N = 33$ and $N = 65$, respectively.

For the analytical engineering problems, i.e., *Three-Bar Truss*, *Cantilever Beam*, *Helical Spring* and *Pressure Vessel*, we used a similar ratio of initial sampling points per design variables as in the analytical benchmarks, i.e., $N = 21$, $N = 21$, $N = 33$ and $N = 45$, respectively.

For the computer simulation model, i.e., *Car Impact*, we used $N = 60$ as initial DOE. The software HyperStudy (part of HyperWorks suite) was used to automate the models parametrization and run.

In order to measure the quality of fit for the analytical problems (R^2 and $NRMSE$), $N_{test} = 5000$ points were created by using the Latin Hypercube Matlab function `lhsdesign`, optimized with `maxmin` criterion set to 100 iterations.

We will not repeat here the validation process summarized in Fig. 31. We already demonstrated that the optimum solution is quite similar for different initial DOE. Even though, for all analytical problems, we run ten times, with different initial DOE, and we present here only the best solution achieved in each case.

In case of the *Car Impact* problem, since the number of points that can be generated is time demanding, we run the problem only once and the respective results are presented.

9 Results and discussion

"Don't torture the data.

If you torture the data long enough, it will confess."

Ronald Coase, 1910-2013.

In this section we will present and discuss the results of the numerical experiments performed in **Phase III** of the research to verify implementation of LSEGO for handling constrained optimization problems, as we detailed in Chapter 8.

9.1 Analytical Benchmark Functions

9.1.1 Braninh-Hoo

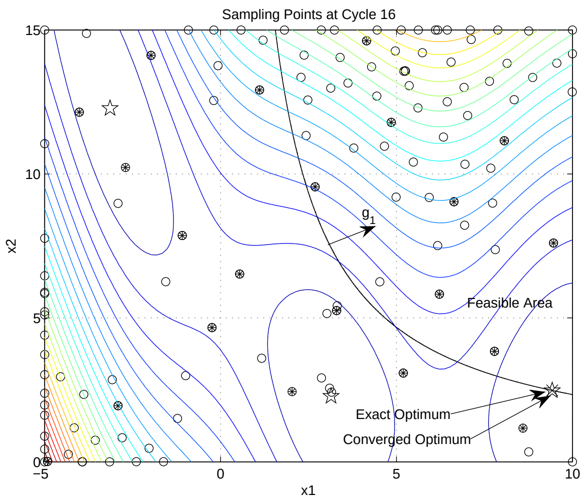


Figure 34: Optimization results for the constrained Branhin-Hoo function. LSEGO-10 converged exactly to the constrained optimum at $\mathbf{x}^* = (9.425, 2.475)$ after 15 cycles. Note the higher density and uniformity of infill points inside than outside the feasible area. The circles "o" are the infill sampling points and the initial DOE points are labeled with asterisks "$*$". The three unconstrained local optima of Branhin-Hoo are labeled as "stars".

For the Branhin-Hoo function, the results of the constrained optimization with LSEGO-10 are presented in Fig. 34. In this case, with the constraint defined with Eq. (28), there is only one global optimum in the feasible area, exactly at the constraint boundary at

$$\mathbf{x}^*_{exact} = (9.425, 2.475)\,.$$

Note that the algorithm added infill points at the whole design space, but the density and uniformity of infill points inside is higher than outside of the feasible area.

The evolution of the objective function $y(\mathbf{x})$ and the normalized constraint $g(\mathbf{x})$ during the optimization cycles with LSEGO-10 for Branin-Hoo function are presented in Fig. 35. Note that LSEGO-10 started very far from the global minimum $y^* = 16.4266$ and reached fast the neighborhood of $\mathbf{x}^*_{exact}$, at the second optimization cycle, with $y^* = 0.5987$, although no further improvement on $y(\mathbf{x})$ was observed until cycle 14. LSEGO-10 converged exactly to the global optimum at cycle 15.

9.1.2 Hartman-3

In the same way, the results for Hartman-3 function is presented in Fig. 36. LSEGO-10 reached the neighborhood of $\mathbf{x}^*_{exact}$ at cycle 5, with $y^* = -3.7899$, i.e., 1.89% error from y^*_{exact}.

The algorithm evolved in the cycles, by reducing the value of $g(\mathbf{x})$ and trying to reach the constraint boundary (i.e., $g(\mathbf{x}) = 0$). At cycle 15, it was found $y^* = -3.8371$, or 0.66% error from y^*_{exact}. At cycle 31, the algorithm reached

$$\mathbf{x}^* = (0.1444, 0.5553, 0.8537)\,,$$

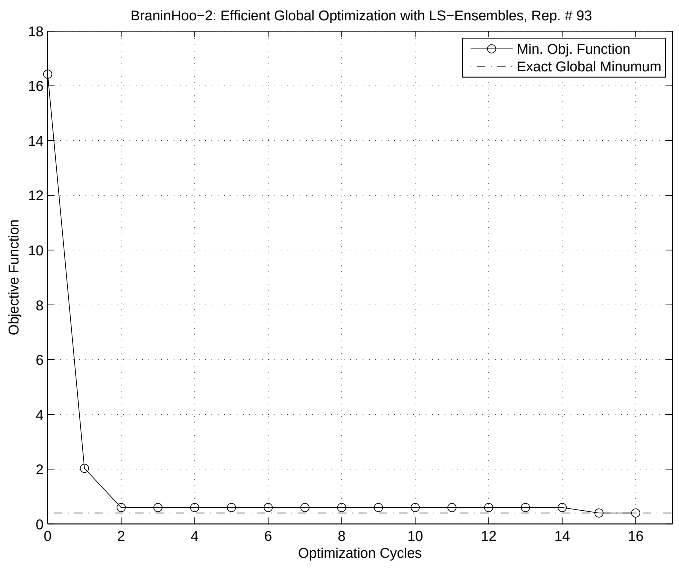
with $y^* = -3.8621$ or 0.02% error from y^*_{exact}, and no further improvement in $y(\mathbf{x})$ or $g(\mathbf{x})$ was found up to 40 cycles.

9.1.3 Hartman-6

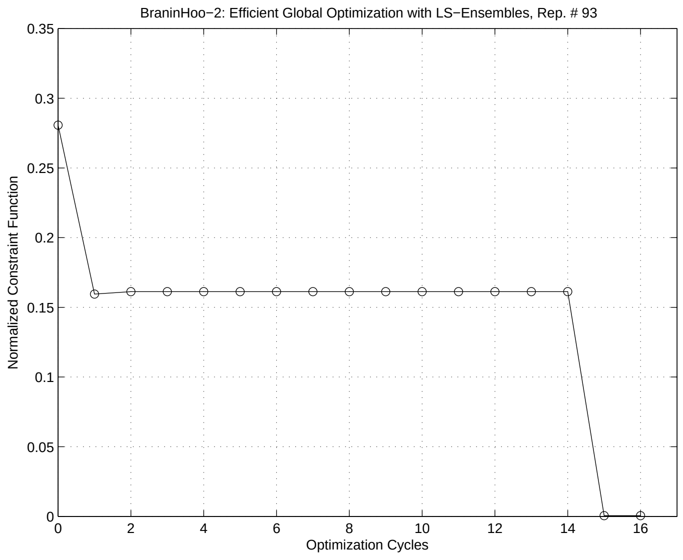
In case of Hartman-6 function, see Fig. 37, the algorithm converged at a slower rate to the neighborhood of $\mathbf{x}^*_{exact}$. At cycle 35, $y^* = -3.100329$, or 6.68% error from y^*_{exact}. At cycle 38, $y^* = -3.177094$, or 4.37% error from y^*_{exact}. We stopped the algorithm at cycle 50 with little improvement with respect to cycle 38, i.e.,

$$\mathbf{x}^* = (0.2688, 0.1547, 0.4508, 0.2890, 0.3433, 0.6541)\,,$$

and $y^* = -3.2082$, or 3.44% error from y^*_{exact}.

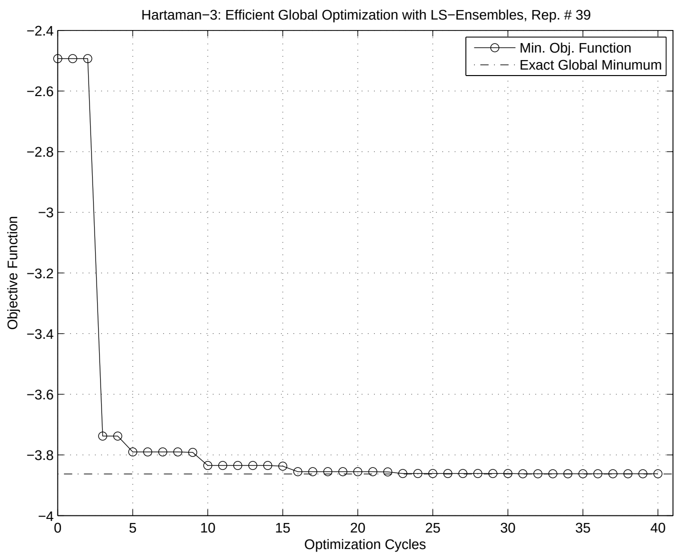


(a) Objective Function $y(\mathbf{x})$

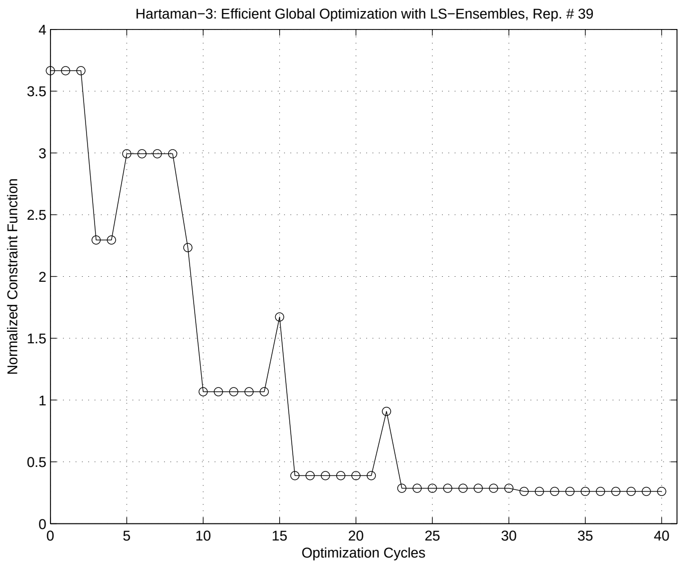


(b) Normalized Constraint $g(\mathbf{x})$

Figure 35: Evolution of the objective function $y(\mathbf{x})$ and the normalized constraint $g(\mathbf{x})$ during the optimization cycles with LSEGO-10 for Branin-Hoo function. LSEGO-10 started with $y^* = 16.4266$ and reached $y^* = 0.5987$ at cycle 2. LSEGO-10 converged to the global optimum with $y^* = 0.3979$ at cycle 15.

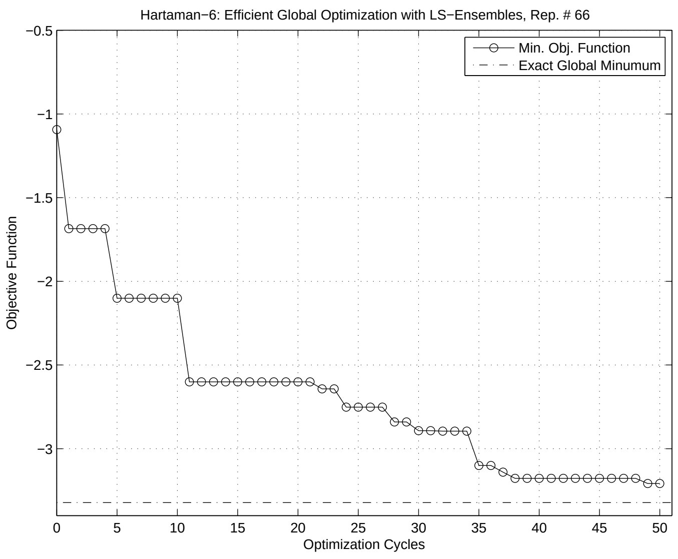


(a) Objective Function $y(\mathbf{x})$

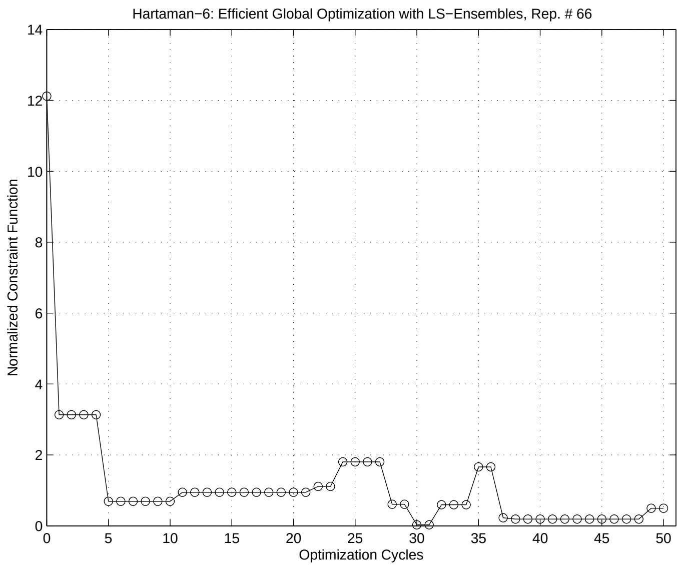


(b) Normalized Constraint $g(\mathbf{x})$

Figure 36: Evolution of the objective function $y(\mathbf{x})$ and the normalized constraint $g(\mathbf{x})$ during the optimization cycles with LSEGO-10 for Hartman-3 function. LSEGO-10 found $y^* = -3.7899$, i.e., 1.89% error from y^*_{exact} at cycle 5. The algorithm converged at cycle 31 with $y^* = -3.8621$ or 0.02% error from y^*_{exact}.



(a) Objective Function $y(\mathbf{x})$



(b) Normalized Constraint $g(\mathbf{x})$

Figure 37: Evolution of the objective function $y(\mathbf{x})$ and the normalized constraint $g(\mathbf{x})$ during the optimization cycles with LSEGO-10 for Hartman-6 function. At cycle 35, $y^* = -3.100329$, or 6.68% error from y^*_{exact}. The algorithm was stopped at cycle 50 with little improvement with respect to cycle 38. At this point $y^* = -3.2082$, or 3.44% error from y^*_{exact}.

9.1.4 Summary for Analytical Benchmarks

We repeated these numerical experiments with the analytical benchmark functions and the convergence pattern was nearly the same for different initial sampling points. It is worth noting that in all the cases, the optimization algorithm presented the convergence behavior in steps. Observe this fact in Fig. 35 for Branin-Hoo and with more pronounced effect in Fig. 36 for Hartman-3 and in Fig. 37 for Hartman-6 function.

As discussed in Forrester and Keane (2009), this stepwise behavior can be understood as the algorithm "switching" from the exploitation to exploration modes during the optimization cycles. In other words, after adding some infill points in the beginning cycles, the quality of fit of the metamodels increases and the algorithm is able to find some improvement in the objective function (exploitation mode).

In the sequence, the algorithm switches to the exploration mode at some cycles, and the next downhill "jump" in direction to the optimum is only achieved after the quality of fit of the metamodels of $y(\mathbf{x})$ and $g(\mathbf{x})$ is enough to promote the next improvement. If we track the quality of fit of the objective function during the optimization cycles, this behavior can be observed.

See for instance the stepwise evolution of the quality of approximation of $y(\mathbf{x})$ by means of NRMSE and R^2 during the optimization cycles for Hartman-3 function in Fig. 38. In this case, at the initial cycles, the quality of fit is irregular/erratic at some extent, with jumps at each three or five steps (cycles 3, 5, 10, 15, ...).

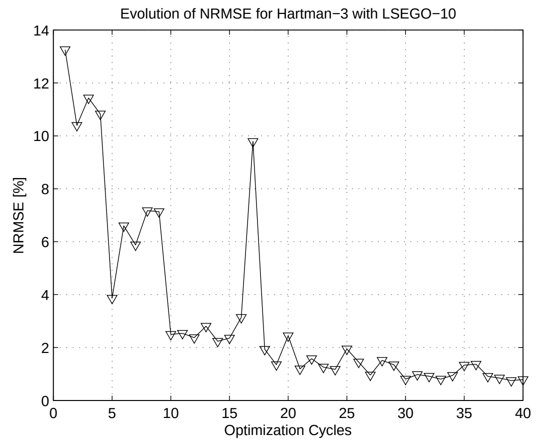
Recall to Fig. 36 and note that these jumps occur simultaneously to the ones observed for the main improvements in objective function and for the constraint. When the accuracy of the metamodel reaches stable levels (i.e., $R^2 > 0.9$ and $NRMSE < 2\%$, around cycle 20, the algorithm is quite close to the global optimum and it converges at cycle 25, when the quality of fit is very good (i.e., $R^2 \approx 1$).

In this sense, based on this observed behavior for the algorithm, it is recommended in practical applications to monitor the quality of fit for the metamodels in parallel to the evolution of the objective and constraint functions, in order to avoid premature or false convergence at suboptimal points.

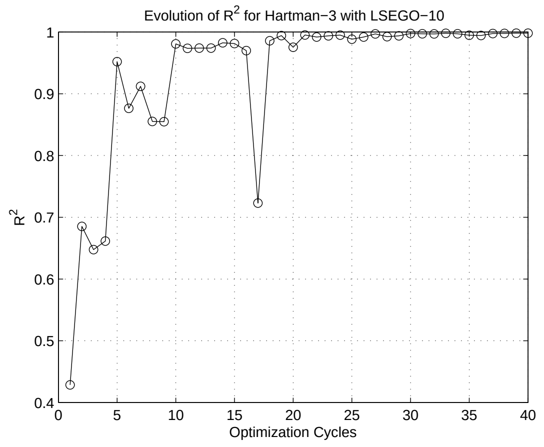
As we discussed before in Section 6.3, in practical situations, this balance between quality of fit, improvement of objective function and constraints versus total number of sampling points, must be observed for each problem.

In addition, in several practical problems, improvements in the objective or constraints of orders of 10% to 25% are very hard to meet. In such situations, finding one or a set of truly improved designs is much more important than finding the "global" optimum for the problem, and the decision on when to stop the optimization cycles must be taken based on these considerations as well.

These results with the analytical benchmarks showed that LSEGO algorithm works successfully to handle constrained optimization problems as well. A deep numerical investigation must be performed with different functions (number of variables, nonlinearity, multimodality, etc.) and increasing number of constraints, in order to understand in detail the behavior, convergence properties, advantages and limitations of LSEGO algorithm.



(a) NRMSE, Hartman-3



(b) R^2, Hartman-3

Figure 38: Evolution of the quality of fit of $y(\mathbf{x})$, by means of NRMSE and R^2 during the optimization cycles with LSEGO-10 for Hartman-3 function. The optimization algorithm switches from exploitation to the exploration mode during the cycles and the convergence is achieved in steps. The downhill "jumps" in direction to the optimum occur in steps, after the quality of fit of the metamodels of $y(\mathbf{x})$ and $g(\mathbf{x})$ is enough to promote the improvement.

9.2 Analytical Engineering Problems

9.2.1 Three-Bar Truss

In Fig. 39 and Fig. 40 it is presented the evolution of the objective functions $f_1(\mathbf{x})$ (weight) and $f_2(\mathbf{x})$ (displacement) and also the respective normalized stress constraints for the *Three-Bar Truss* problem with two variables and three constraints, as detailed in Eq. (29).

Note in Fig. 40 that LESEGO-10 converged exactly to the reference optimum, after two cycles for $f_2(\mathbf{x})$ (displacement). On the other hand, although with significant improvement in the objective function, the algorithm did not find any improved solution after cycle 11 for $f_1(\mathbf{x})$ (weight).

For the *Three-Bar Truss* problem, we forced intentionally the algorithm to start in a region completely away from the exact solution, in order to see if the convergence can still be achieved with a very poor initial DOE. Observe in Fig. 41 that for $f_1(\mathbf{x})$ it was considered $(4 \leq x_{1,2} \leq 5)$ and for $f_2(\mathbf{x})$ it was considered $(1 \leq x_{1,2} \leq 1)$ as region for the initial DOE.

In case of $f_1(\mathbf{x})$ (weight) for the *Three-Bar Truss* problem, we applied a sequential clustering procedure to see how LSEGO-10 behaves and if it is possible to accelerate and improve the convergence in direction of the exact solution of reference.

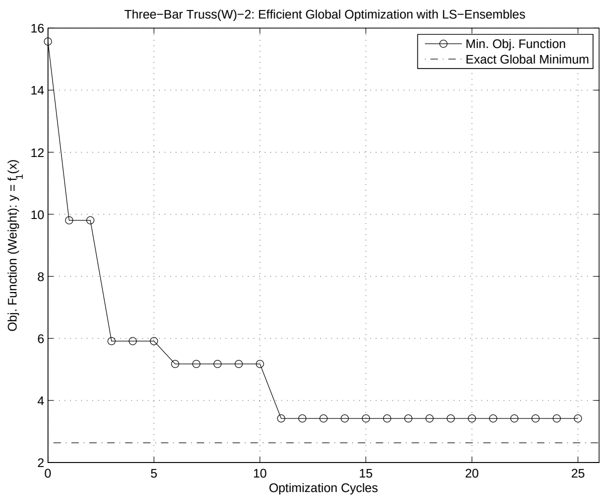
We restarted the optimization after cycle 08 by reducing the design space to $(0 \leq x_{1,2} \leq 2)$ and by adding ten extra points generated with Matlab function `lhsdesign`, see Fig. 43.

After the first cluster, the solution improved significantly at cycle 10 to the point

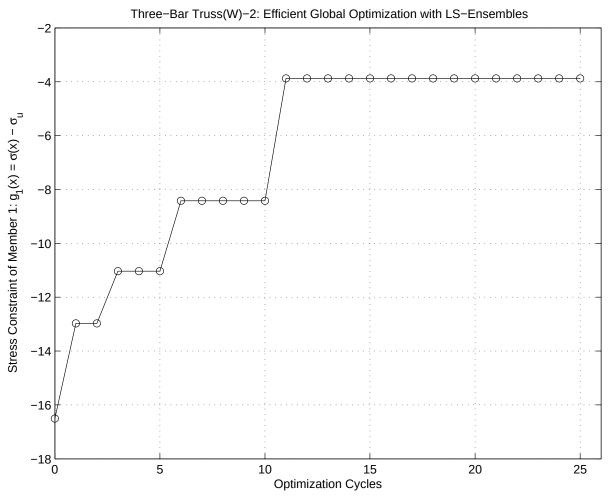
$$\mathbf{x}^* = (0.9008, 0.1707),$$

with $f_1^* = 2.7186$ (or 3.23% error from $f_{1-exact}^*$).

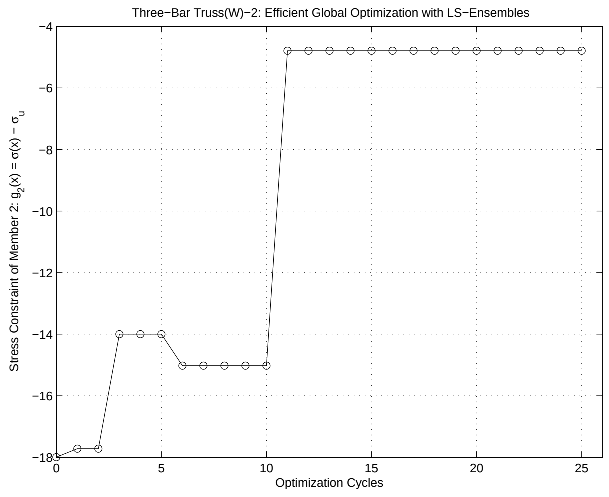
Again, the algorithm stalled at the same optimum for more two cycles. Then we applied a second clustering procedure after cycle 12, with another additional ten points and the design space set to $(0 \leq x_{1,2} \leq 1)$ in the optimization. This second cluster produced no effect and we stopped the optimization at cycle 15. See the overall picture in Fig. 42, with the design space at the end of cycle 15.



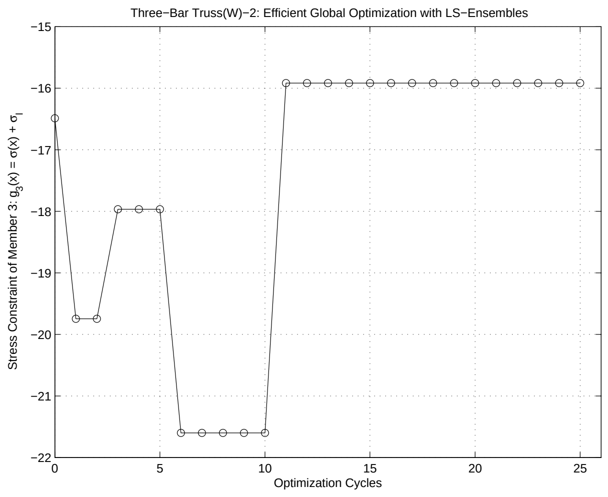
(a) Objective Function $f_1(\mathbf{x})$



(b) Normalized Constraint $\sigma_1(\mathbf{x})$

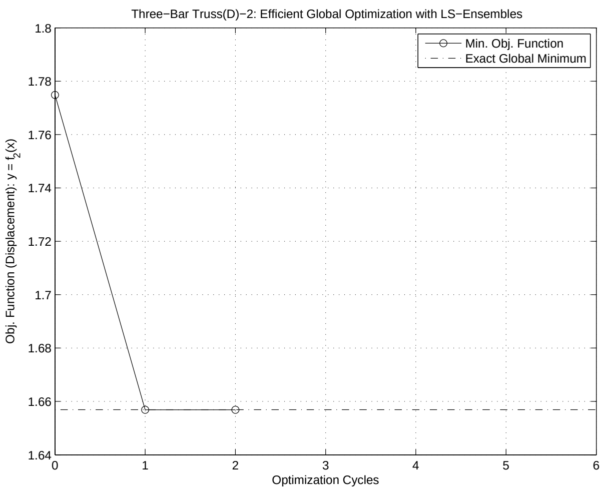


(c) Normalized Constraint $\sigma_2(\mathbf{x})$

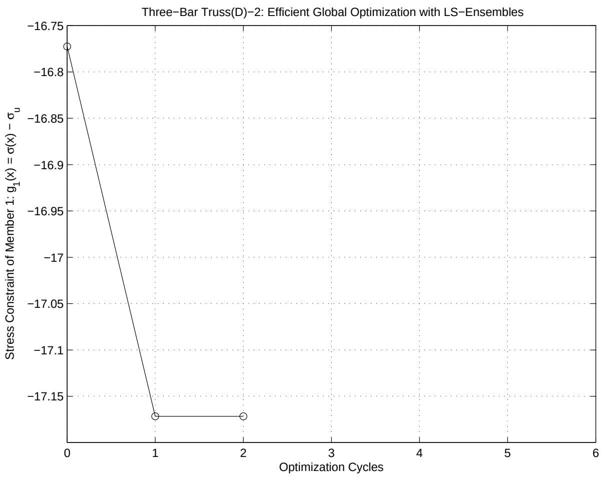


(d) Normalized Constraint $\sigma_3(\mathbf{x})$

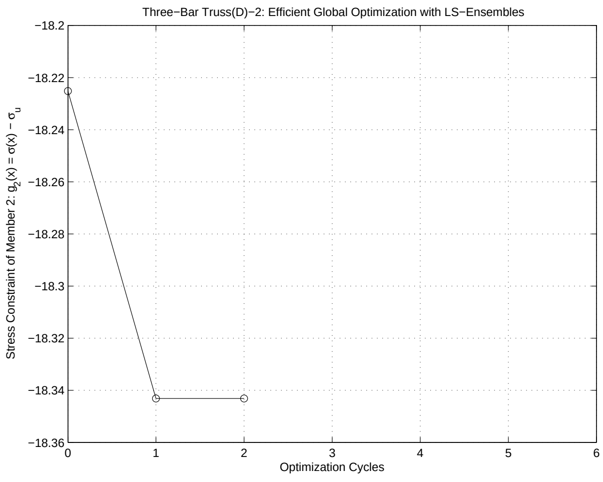
Figure 39: Evolution of the objective function $f_1(\mathbf{x})$ (weight) and the normalized stress constraints $\sigma_1(\mathbf{x})$, $\sigma_2(\mathbf{x})$ and $\sigma_3(\mathbf{x})$ during the optimization cycles with LSEGO-10 for the *Three-Bar Truss* problem.



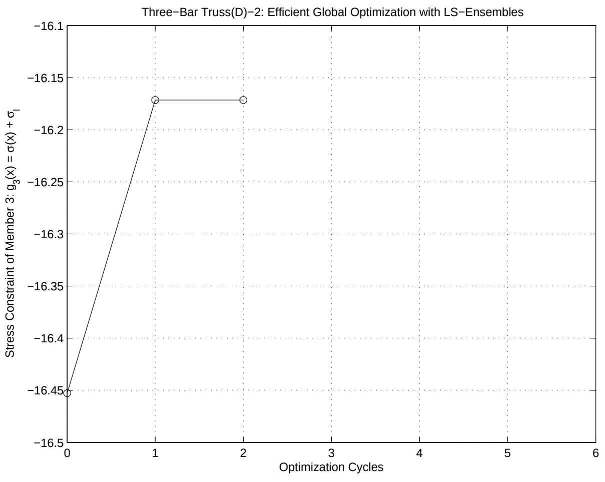
(a) Objective Function $f_1(\mathbf{x})$



(b) Normalized Constraint $\sigma_1(\mathbf{x})$

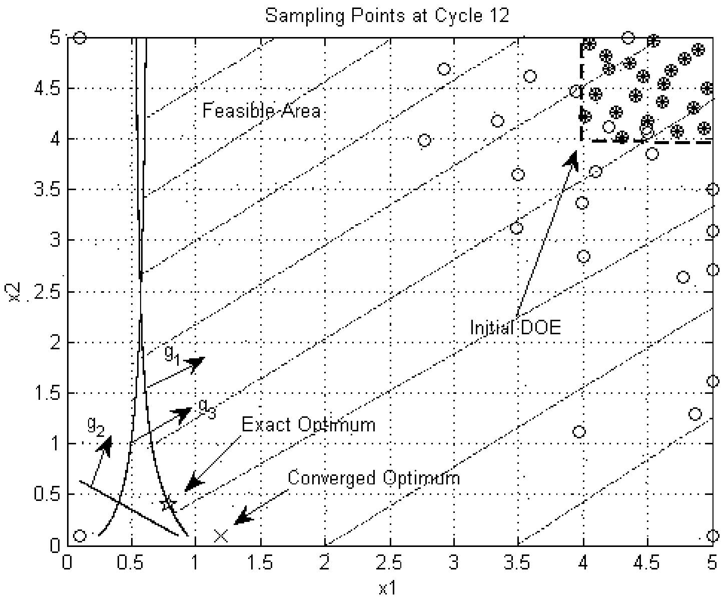


(c) Normalized Constraint $\sigma_2(\mathbf{x})$

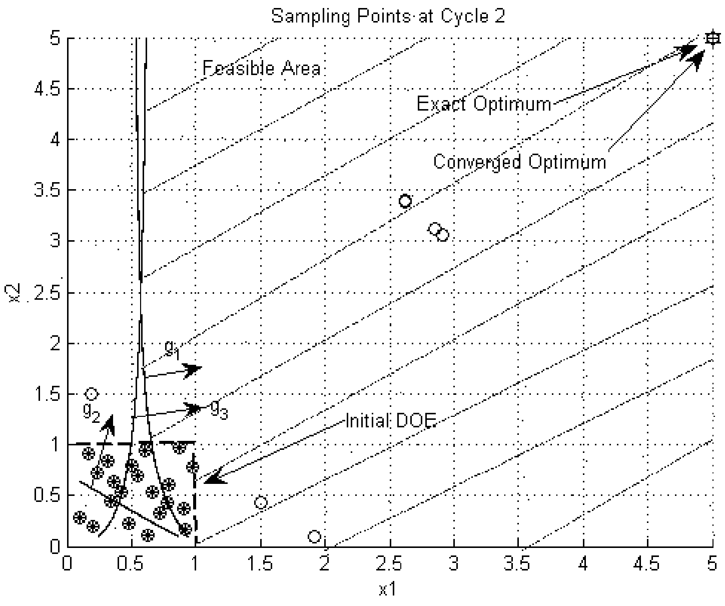


(d) Normalized Constraint $\sigma_3(\mathbf{x})$

Figure 40: Evolution of the objective function $f_2(\mathbf{x})$ (displacement) and the normalized stress constraints $\sigma_1(\mathbf{x})$, $\sigma_2(\mathbf{x})$ and $\sigma_3(\mathbf{x})$ during the optimization cycles with LSEGO-10 for the *Three-Bar Truss* problem.

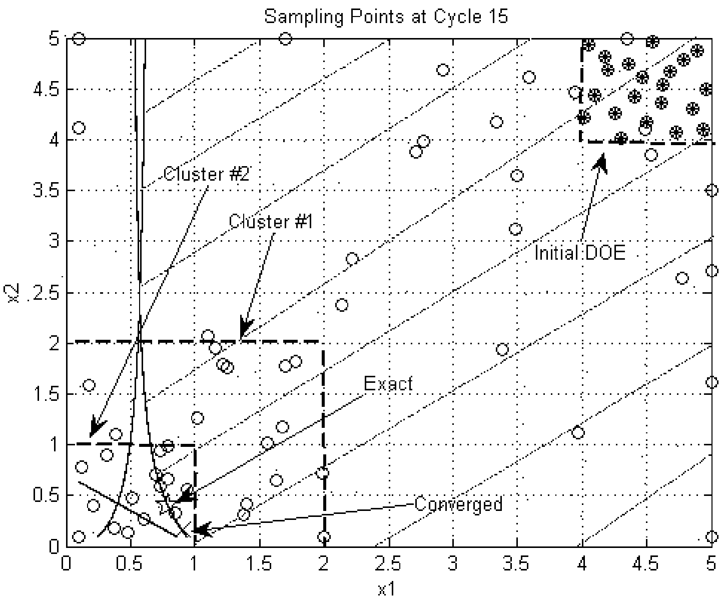


(a) Design Space for $f_1(\mathbf{x})$



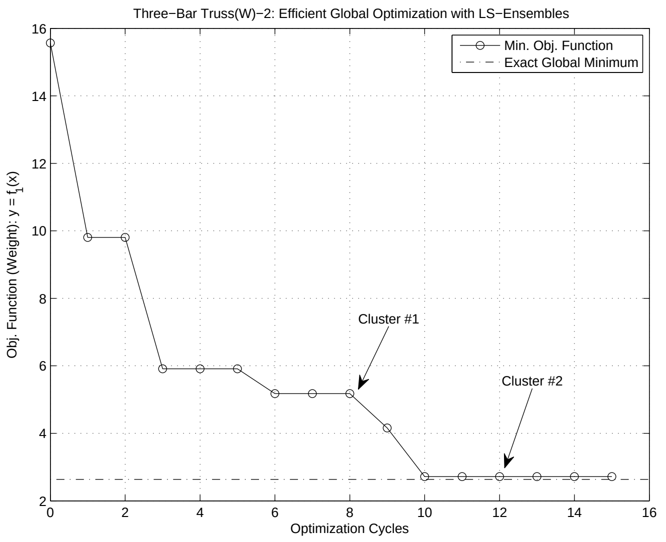
(b) Desigin Space for $f_2(\mathbf{x})$

Figure 41: Design space for $f_1(\mathbf{x})$ (weight) and $f_2(\mathbf{x})$ (displacement) at convergence for the *Three-Bar Truss* problem. Feasible area indicated by hatches. Even starting the initial DOE far away from the exact solution, LSEGO-10 was able to converge to exactly or at the optimum neighborhood in few optimization cycles.



(a) Design Space $f_1(\mathbf{x})$

Figure 42: Design space for $f_1(\mathbf{x})$ (weight) at convergence for the *Three-Bar Truss* problem, after two clusters at cycle 08 and cycle 12. Feasible area indicated by hatches. After the clustering procedure, LSEGO-10 converged to $\mathbf{x}^* = (0.9008, 0.1707)$ with $f_1^* = 2.7186$ (or 3.23% error from $f^*_{1-exact}$).



(a) Objective Function $f_1(\mathbf{x})$

Figure 43: Evolution of the objective function $f_1(\mathbf{x})$ (weight) for the *Three-Bar Truss* problem with a sequential clustering procedure at cycle 08 and cycle 12. After the clustering procedure, LSEGO-10 converged to $\mathbf{x}^* = (0.9008, 0.1707)$ with $f_1^* = 2.7186$ (or 3.23% error from $f^*_{1-exact}$).

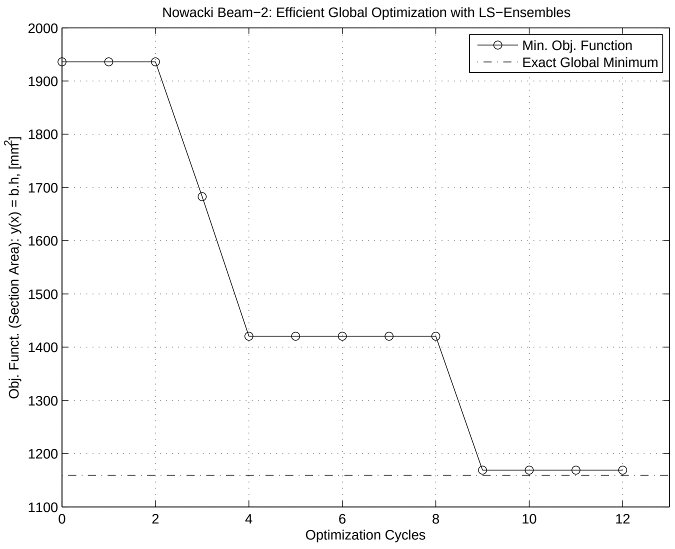
9.2.2 Cantilever Beam

In Figs. 44 and 45 it is presented the evolution of the optimization cycles for the *Cantilever Beam* problem with two variables and two constraints, detailed in Eq. (30).

In this case, LSEGO-10 converged almost exactly at cycle 09 to the point

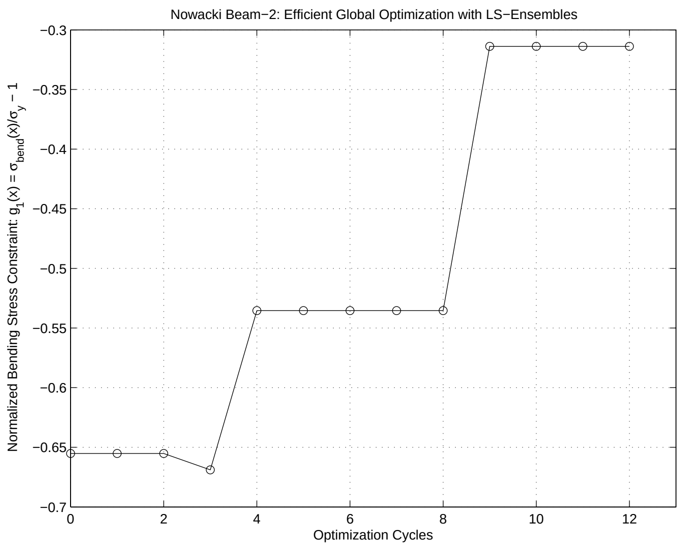
$$\mathbf{x}^* = (5.0000, 233.7748),$$

with $y^* = 1168.8739$ (or 0.33% error from y^*_{exact}) and no further improvement was observed until cycle 12.

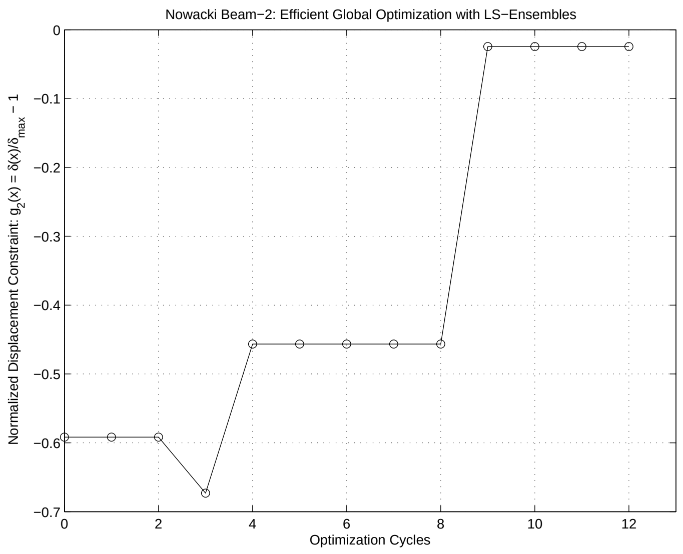


(a) Objective Function $y(\mathbf{x})$

Figure 44: Evolution of the objective function $y(\mathbf{x})$, during the optimization cycles with LSEGO-10 for the *Cantilever Beam* problem. LSEGO-10 converged almost exactly at cycle 09 to the point $\mathbf{x}^* = (5.0000, 233.7748)$ with $y^* = 1168.8739$ (or 0.33% error from y^*_{exact}.)



(a) Constraint Function $\delta(\mathbf{x})$



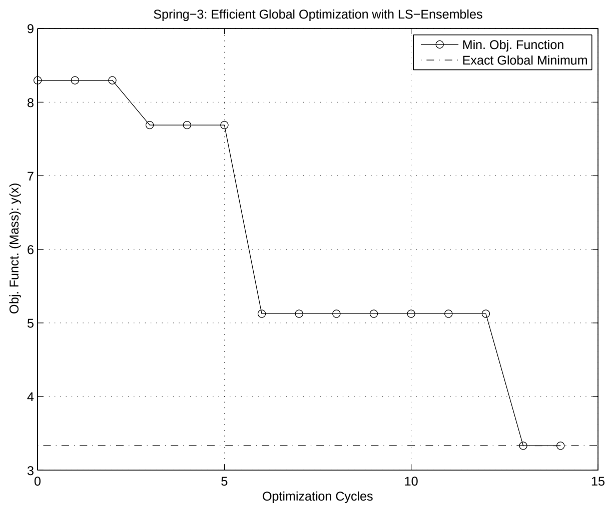
(b) Constraint Function $\sigma(\mathbf{x})$

Figure 45: Evolution of the normalized constraints $\delta(\mathbf{x})$ (displacement) and $\sigma(\mathbf{x})$ (bending stress), during the optimization cycles with LSEGO-10 for the *Cantilever Beam* problem. LSEGO-10 converged almost exactly at cycle 09 with 0.33% error from y^*_{exact}, and no further improvement was observed for objective and constraints until cycle 12.

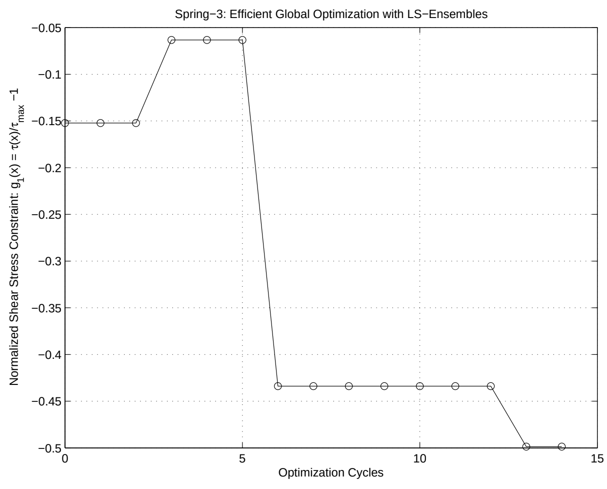
9.2.3 Helical Spring

The results for the *Helical Spring* problem with three variables and three constraints, detailed in Eq. (31), are presented in Fig. 46.

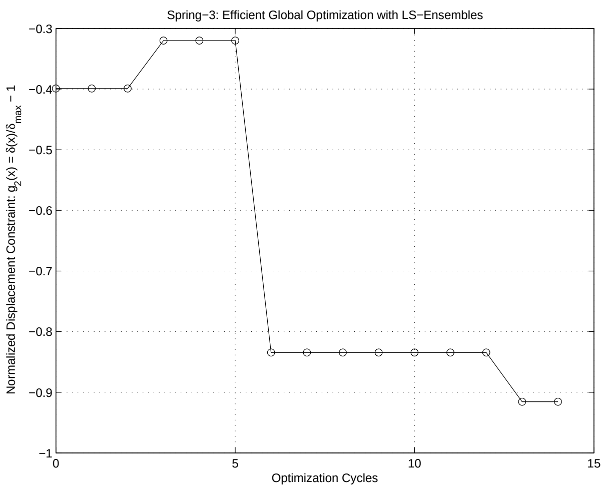
In this case, LSEGO-10 converged to the exact solution at cycle 13.



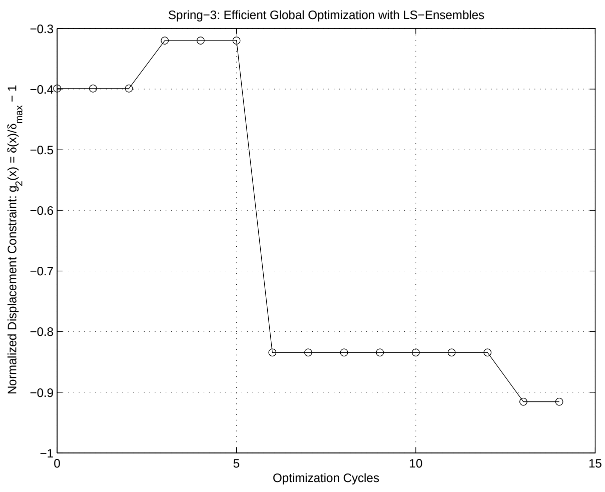
(a) Objective Function $y(\mathbf{x})$



(b) Constraint Function $\delta(\mathbf{x})$



(c) Constraint Function $\tau(\mathbf{x})$



(d) Constraint Function $F_n(\mathbf{x})$

Figure 46: Evolution of the objective function $y(\mathbf{x})$ (mass) and the normalized constraints $\delta(\mathbf{x})$ (displacement), $\tau(\mathbf{x})$ (shear stress) and $F_n(\mathbf{x})$ (natural frequency), during the optimization cycles with LSEGO-10 for the *Helical Spring* problem. LSEGO-10 converged to the exact solution at cycle 13.

9.2.4 Pressure Vessel

In Fig. 47 it is presented the evolution of the optimization cycles for the *Pressure Vessel* problem, with four variables and four constraints, detailed in Eq. (32).

In same way of $f_1(\mathbf{x})$ (weight) for the *Three-Bar Truss* problem, the algorithm improved the objective function but stalled at a suboptimal solution at cycle 15 and no further improvement was found up to cycle 34.

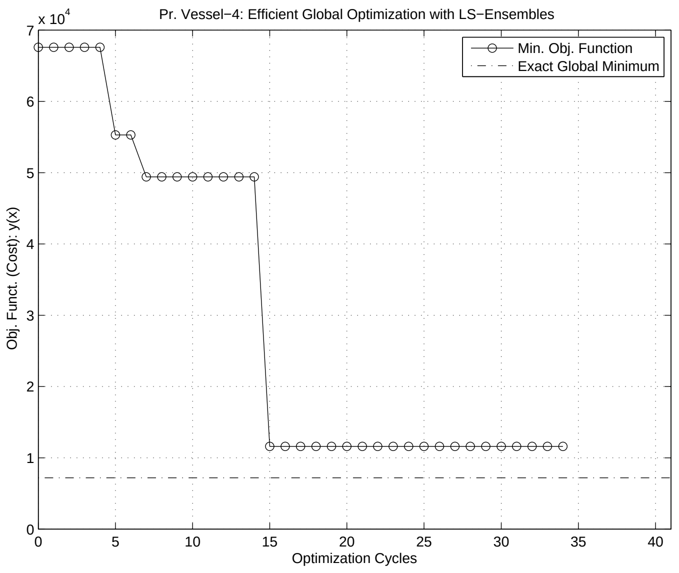
We applied as well a sequential clustering procedure to see how LSEGO-10 behaves and to verify if it is also possible to accelerate and improve the convergence in direction of the reference exact solution.

After the clustering procedure, see Fig. 48, LSEGO-10 converged at cycle 19 to the point

$$\mathbf{x}^* = (1.0000, 0.4375, 45.9244, 152.9694),$$

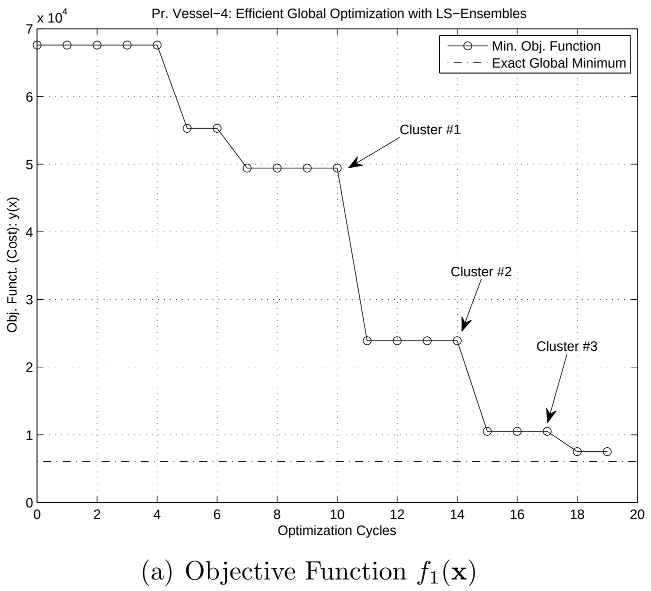
with $y^* = 7408.5038$, with a total of 473 evaluations of the true/exact function $f(\mathbf{x})$. In Fig. 49, the evolution of the constraints after the clustering procedure is presented.

This solution is 2.92% different from the first solution reported by Kannan and Kramer (1994). If we compare with the best value presented in Yang et al (2013a), the difference is 22.26%, but in this case the solution was found by using evolutionary computation, at cost of an order of 80×10^3 evaluations of the true/exact function $f(\mathbf{x})$, what is obviously non feasible in the context of metamodel based optimization.



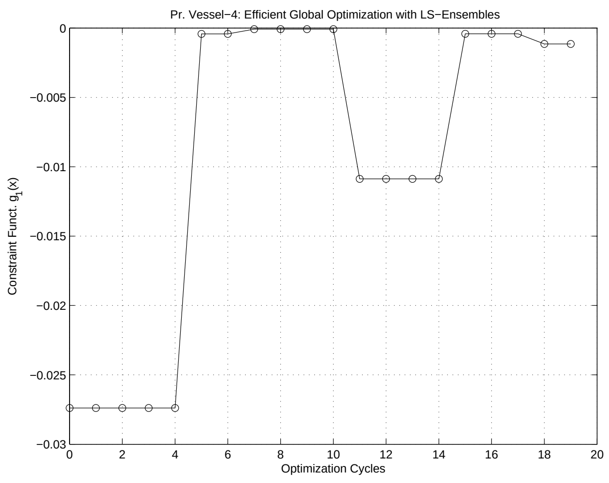
(a) Objective Function $f_1(\mathbf{x})$

Figure 47: Evolution of the objective functions $y(\mathbf{x})$ during the optimization cycles with LSEGO-10 for the *Pressure Vessel*. The algorithm converged to a suboptimal at cycle 15 and no further improvement was found up to cycle 34.

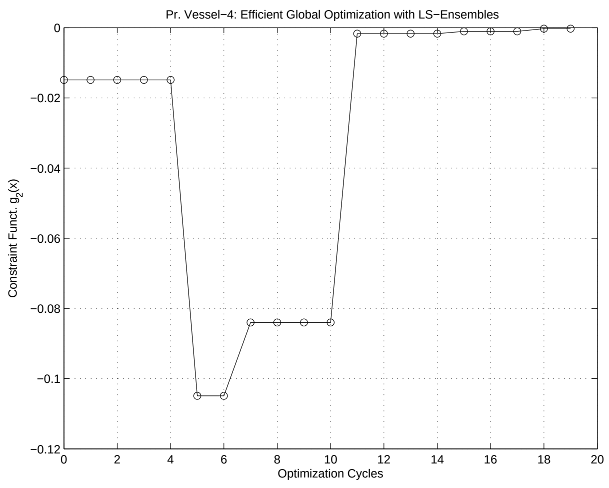


(a) Objective Function $f_1(\mathbf{x})$

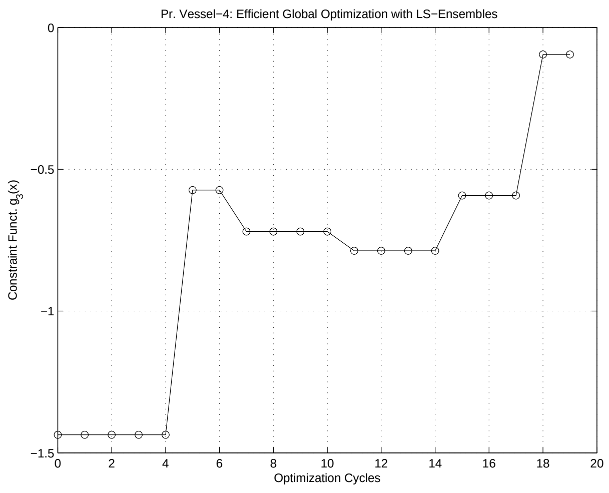
Figure 48: Evolution of the objective functions $y(\mathbf{x})$ during the optimization cycles with LSEGO-10 for the *Pressure Vessel* problem, with the clustering procedure.



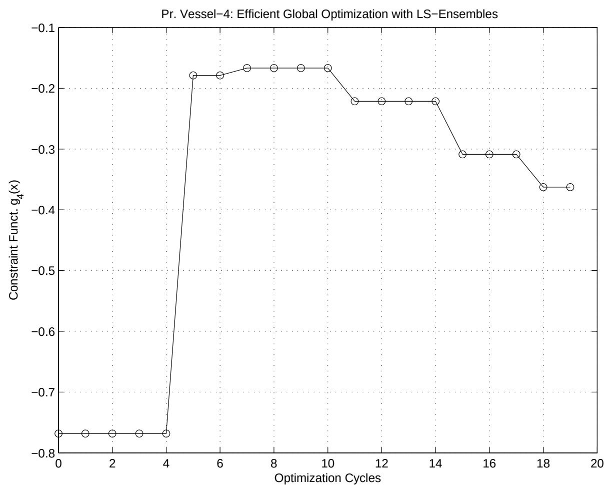
(a) Constraint Function $g_1(\mathbf{x})$



(b) Constraint Function $g_2(\mathbf{x})$



(c) Constraint Function $g_3(\mathbf{x})$



(d) Constraint Function $g_4(\mathbf{x})$

Figure 49: Evolution of the constraint functions $g_1(\mathbf{x})$, $g_2(\mathbf{x})$, $g_3(\mathbf{x})$ and $g_4(\mathbf{x})$ during the optimization cycles with LSEGO-10 for the *Pressure Vessel* problem, with the clustering procedure at cycles 10, 14 and 17.

9.2.5 Sequential Sampling vs. One-Stage Approach

As we discussed in Section 6, there is still some controversy regarding the effectiveness of sequential sampling versus one-stage approaches, ref. Jin et al (2002) and Viana et al (2010).

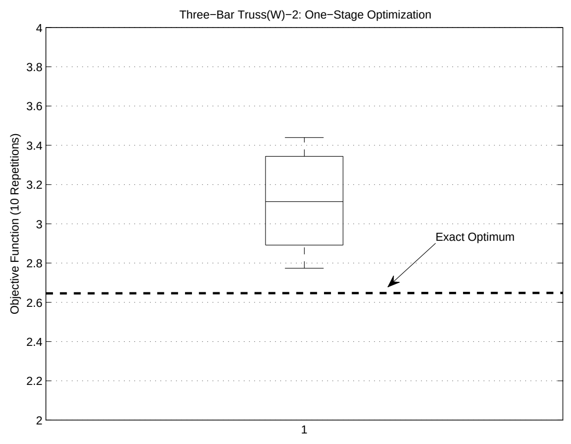
In order to check this fact for our examples with analytical engineering functions, we repeated the one-stage optimization ten times, with different initial DOE, at a very large rate of number of sampling points in terms of number of variables[12], i.e., for $f_1(\mathbf{x})$ of *Three-Bar Truss* $N = 120$ $(60n_{nv})$; for *Cantilever Beam* $N = 120$ $(60n_{nv})$; for *Helical Spring* $N = 360$ $(120n_{nv})$ and for *Pressure Vessel* $N = 460$ $(120n_{nv})$.

The results for this experiment are presented in Fig. 50. For the cases investigated, the results showed that there is no guarantee to achieve the exact optimum with a one-stage approach, even starting the optimization with a high density of sampling points in the design space. Probably, the majority of these points are working only for improving the overall quality of the metamodels (exploration) and these points are not being effective to help finding the exact minimum (exploitation), what is clearly a waste of resources for optimization objectives in mind.

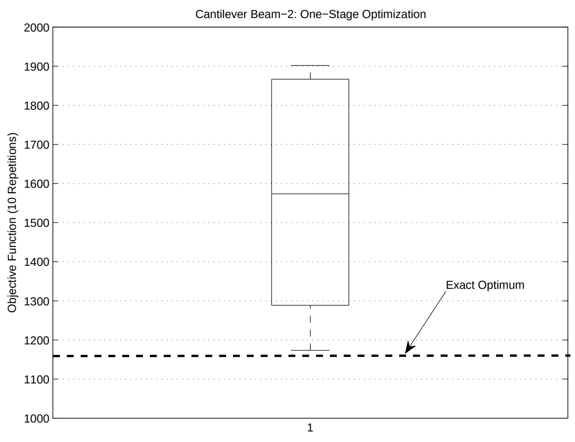
These results confirm our beliefs that it is worthwhile to apply sequential sampling approaches like EGO-type algorithms, or some hybrid approach (allied to clustering, for instance), in order to increase the number of points slowly and "surgically" at regions of the design space, with real chance or expectation of improvement in the objective and constraint responses.

In this sense, we reinforce the comments of Forrester and Keane (2009), that the metamodel based optimization must always include some form of iterative search and repetitive infill process to ensure the accuracy in the areas of interest in the design space. In this direction, we agree on the recommendations that a reasonable number of points for starting the sequential sampling metamodel based optimization is about one third (33%) of the available budget in terms of true function/model evaluations (or processing time) to be spent in the whole optimization cycle.

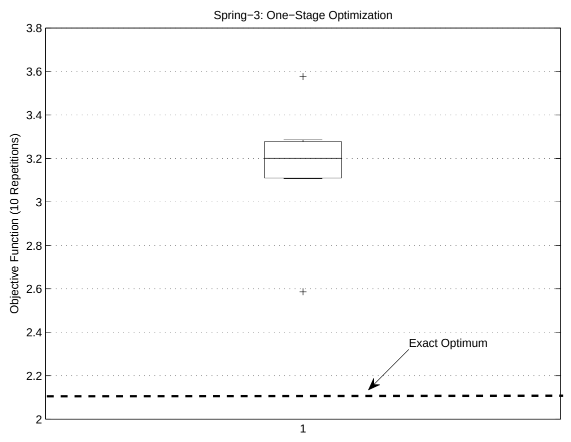
[12] As a common practice for metamodel based optimization purposes, the number of points in initial DOE is often in the range $5n_{nv}$ to $10n_{nv}$.



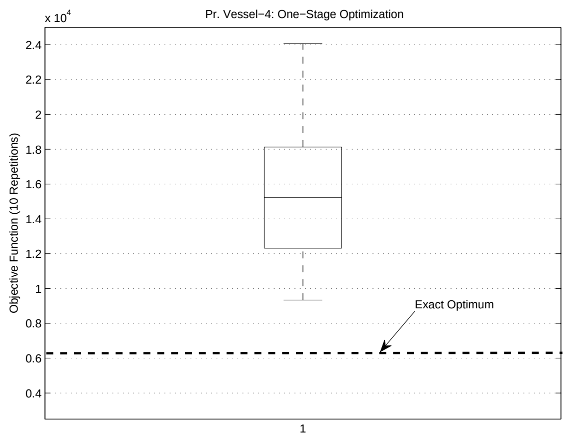
(a) *Three-Bar Truss*



(b) *Cantilever Beam*



(c) *Helical Spring*



(d) *Pressure Vessel*

Figure 50: Boxplots with the converged results for analytical engineering functions with one-stage optimization. Each problem was repeated 10 times with different initial DOE, i.e., for $f_1(\mathbf{x})$ of *Three-Bar Truss* $N = 120$ $(60n_{nv})$; for *Cantilever Beam* $N = 120$ $(60n_{nv})$; for *Helical Spring* $N = 360$ $(120n_{nv})$ and for *Pressure Vessel* $N = 460$ $(120n_{nv})$. Even with very dense number of initial sampling points, there is no guarantee of achieving the exact optimum.

9.2.6 Unfeasible Results Near Constraints Boundaries

As can be observed in the results presented here, some optimum found turned out to be unfeasible (Hartman-6) or, although the constraints were not violated in the majority of cases studied in this thesis, it was hard o find the optimum at the desirable location, that is, at a constraint boundary (active constraints).

As discussed by Viana (2011) in constrained optimization (with constraints being metamodels) or in reliability-based optimization, it can happen that after running the optimization the solution found should be infeasible due to metamodel errors.

In order to try to avoid this kind of "pathology", the first thing that can be done is the correct choice of constraints to be included in the optimization, specially the redundant ones and those that are unlikely to be active (at the constraint boundary), as discussed by Forrester et al (2008).

In this cases, some sort of penalization approaches can be applied to the inactive or violated constraints to force the optimum to the boundary or feasible region. In addition, other strategies by managing the samples to favor the boundary or the feasible region can be applied, as remarked by Forrester et al (2008).

Viana (2011) extended this discussion and possible directions can be (i) use of conservative constraints based on margin of safety parameters or targets to push the optimum to the feasible region or (ii) use adaptive sampling methods to improve the prediction capability of the constraints in the boundary between feasible and unfeasible domains.

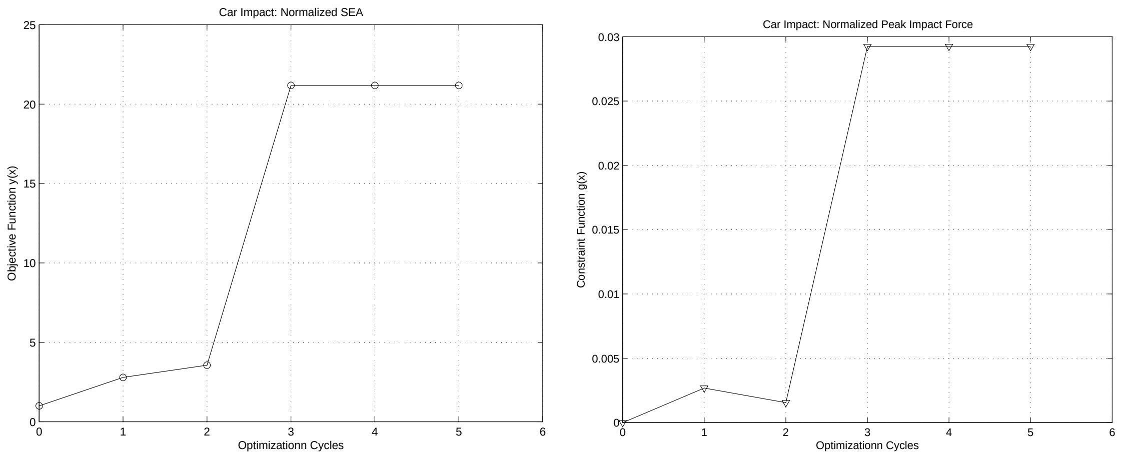
In the present thesis we did not implement any kind of strategy or control for constraint boundary prediction improvement and feasibility assurance. Although it is still an open question, since it is a required feature for any optimization algorithm it is strongly recommended to be studied and implemented in future developments of LSEGO and other metamodel based optimization algorithms.

9.3 Engineering Computer Model: Car Impact

The evolution of objective and constraint functions during the optimization cycles for the *Car Impact* problem is presented in Fig. 51. In this case, LSEGO-10 converged at cycle 03, to the point

$$\mathbf{x}^* = (3.0000, 2.6705, 1.8630, 2.9231),$$

with $y^* = 21.1745$ and $g^* = 0.02924$. In other words, this optimum found by LSEGO-10 has $\approx$ 2000% more specific energy absorption performance, and the maximum impact force was reduced 2.92% with respect to baseline design. We stopped the optimization algorithm at Cycle 05 after no additional improvement in objective or constraint function. The whole process finished with a total of 110 ($27.5n_v$) sampling points (or function evaluations with the true model).



(a) Objective Function $y(\mathbf{x})$ (b) Normalized Constraint $g(\mathbf{x})$

Figure 51: Evolution of the of the objective function $y(\mathbf{x})$ and the constraint function $g(\mathbf{x})$ during the optimization cycles with LSEGO-10 for the Car Impact problem. The algorithm reached the optimum at cycle 03 with with $y^* = 21.1745$ and $g^* = 0.02924$.

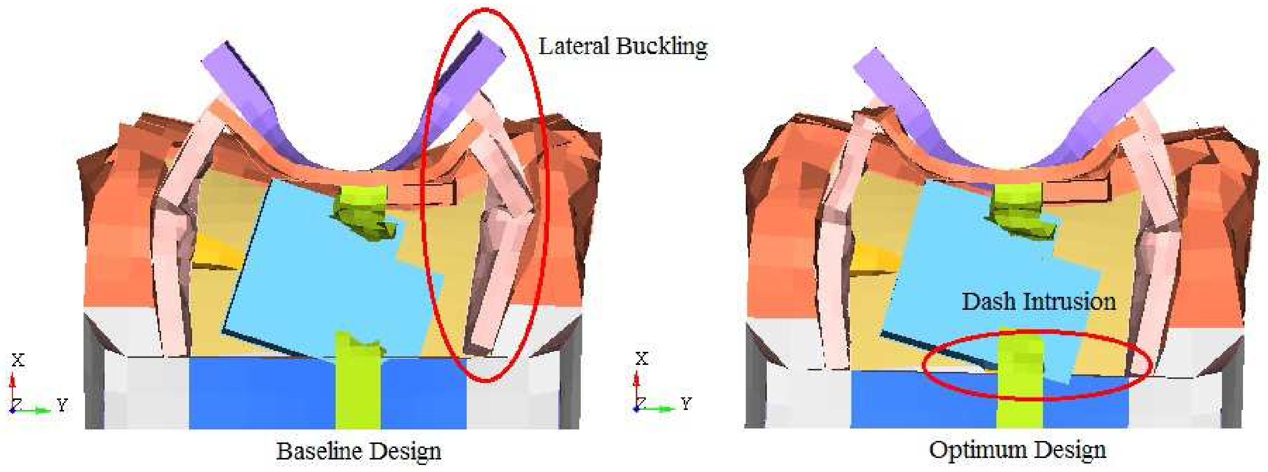
Let us understand with more detail the improvements of the optimum design found with respect to the baseline. See some comparisons in Figs. 52 and 53.

Note in Fig. 52 that the overall lateral buckling of the front end rails is more pronounced for baseline design (right). When lateral buckling occurs, the efficiency of energy absorption due to plastic strains is diminished. In addition, the lateral buckling of the rails tends to reduce the engine compartment available size and, at same time, increase the acceleration of the engine in the direction of the occupant compartment. Therefore, the chance of increasing the engine intrusion through the dash-panel[13] is higher, with more risk of damage to the occupants.

As can be observed in Fig. 52 (left), the intrusion of the engine in the dash-panel region is smaller for the optimum than for the baseline design. In quantitative terms, observe the dash-panel displacements, Fig. 53(a), $\approx$ 37% higher for baseline. In the same way, see in Fig. 53(b) the maximum plastic strain levels in the dash-panel $\approx$ 48% higher for baseline, as compared to the optimum values.

Therefore, based on these qualitative and quantitative comparisons, it is clear that the optimum design found is significantly better in terms of impact performance than the baseline design. In addition, the optimization algorithm converged in only three optimization cycles, with a reasonable cost in terms of total number of sampling points.

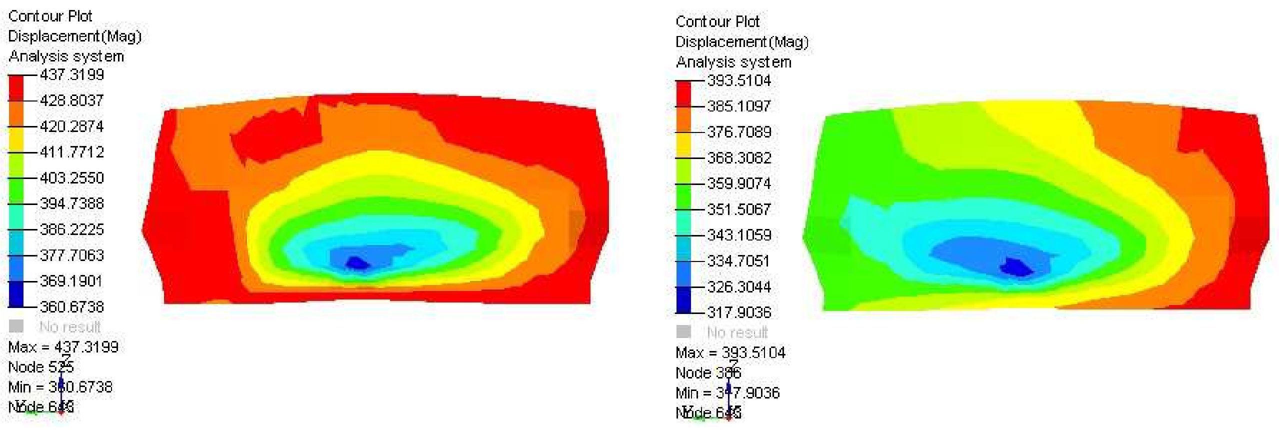
Remember that the objective with this optimization example was the maximization of the specific energy absorbed (SEA), with impact force as constraint (refer to Section 8.2.3). As consequence, the optimum design resulted heavier than the baseline, since the mass was not constrained in the problem statement. If the objective is mass reduction and performance, then another optimization scenarios can be considered, but it is out of the scope of this illustration example.



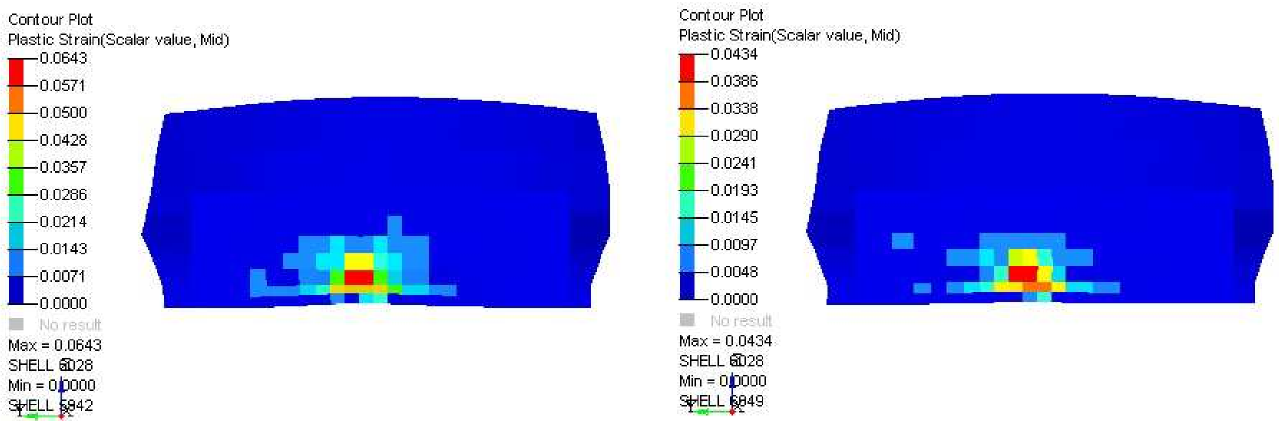
(a) Overall Deformed Shape - Bottom View

Figure 52: Comparison of results for Baseline (left) and Optimum Design (right) for the *Car Impact* problem. Note the more pronounced lateral buckling of the front end rails and also the intrusion of the engine in the dash-panel for the baseline design.

[13]The dash-panel is the main sheet metal part, behind the instrument panel and directly connected to the vehicle floor that separate the cabin or cockpit from the engine compartment.



(a) Dash Panel Displacements - Front View



(b) Dash Panel Plastic Strains - Front View

Figure 53: Comparison of results for Baseline (left) and Optimum Design (right) the *Car Impact* problem. In (a) the maximum dash-panel displacements are ≈ 37% higher for baseline. In (b) the maximum plastic strain levels in the dash-panel are ≈ 48% higher for baseline, as compared to the optimum values.

10 Concluding Remarks

"Don't adventures ever have an end? I suppose not.
Someone else always has to carry on the story."
The Fellowship of the Ring
J.R.R. Tolkien, 1892-1973.

10.1 Summary of Thesis Original Contributions

In this work we reviewed and compiled the results of our previous research in the fields of ensemble of metamodels and efficient global optimization (EGO).

The first part of the research (**Phase I**) is already published in Ferreira and Serpa (2016). In this first paper we presented an approach to create ensemble of metamodels (or weighted averaged surrogates) based on least squares (LS) approximation.

The second part (**Phase II**) has been compiled in a second manuscript. In this second paper, under review by the journal editors and recommended for publication, we presented LSEGO (Least Squares Ensemble EGO), an approach to drive efficient global optimization (EGO), based on the LS (least squares) ensemble of metamodels.

Our previous results with LSEGO were competitive and promising, but they were limited only to unconstrained optimization. In the present thesis we extended the LSEGO algorithm (developed in **Phase I** and **Phase II** to handle constrained optimization problems as well.

Some numerical experiments were performed with analytical benchmark functions and also for an industry scale engineering problem with promising and competitive results.

Our tests were limited to single objective constrained optimization problems with two to four variables and with one to four constraint functions. This extension and results comprise the **Phase III** of our research project and it will summarized and submitted for publication soon.

10.2 Overall Achievements

For all the analytical benchmark and analytical engineering problems investigated, LSEGO was able to converge to the neighborhood of the exact (or reference optimum solution) in few optimization cycles. In some examples, the convergence with LSEGO was slow and the algorithm stalled at suboptimal regions. We observed that, in such situations, the combination of LSEGO with other sequential or adaptive sampling techniques (e.g., design space clustering) can enhance the convergence rate in direction of the reference optimum solution.

In all the problems investigated, LSEGO presented a stepwise convergence pattern, which is common to EGO-type algorithms. In this sense, it is recommended in practical applications to monitor the quality of fit for the metamodels in parallel to the evolution of the objective and constraint functions, in order to avoid premature or false convergence to suboptimal points.

We also investigated the use of one-stage optimization for the analytical engineering functions and this approach was not able to guarantee convergence to the exact solution, even with a very dense initial sampling space. One possible explanation should be that the majority of the initial points design space are working only for improving the overall quality of the metamodels (exploration), and they are not being effective to help finding the exact minimum (exploitation), what is clearly a waste of resources for optimization objectives.

These fact confirms that it is worthwhile to apply sequential sampling approaches like EGO-type algorithms, or some other hybrid approach (allied to clustering, for instance), in order to increase the number of points slowly and "surgically", at potential optimum regions of the design space. Consequently this kind of method can lead to real improvement in the objective and constraint responses, instead of using a one-stage approach.

At the end, we performed a brief case study with the application of LSEGO in the optimization of an engineering problem, i.e., a simplified version for the full Car Frontal Pole Impact test. In this case, LSEGO converged in three optimization cycles to an improved and feasible design. Based on qualitative and quantitative comparisons, we verified that the optimum design found is significantly better in terms of impact performance than the baseline design. In addition, the optimization algorithm converged with a reasonable cost in terms of total number of sampling points.

The results presented in this study showed that the LSEGO algorithm works successfully to handle constrained optimization problems in a feasible number of optimization cycles. On the other hand, a deeper numerical investigation must be performed with different functions (number of variables, nonlinearity, multimodality, etc.) and increasing number of constraints, in order to understand in detail the behavior, convergence properties, advantages and limitations of LSEGO algorithm.

As can be observed in the results presented here, although the constraints were not violated in the majority of cases studied in this thesis, it was hard o find the optimum at constraint boundaries (active constraints).

The inaccuracy of metamodels at constraint boundary is known pathology discussed in the metamodel optimization literature. In the present thesis we did not implement any kind of strategy or control for constraint boundary prediction improvement and feasibility assurance. Although it is still an open question, since it is a required feature for any optimization algorithm it is strongly recommended to be studied and implemented in future developments of LSEGO and other metamodel based optimization algorithms.

10.3 Some Possible Future Directions

At last but not least, it is possible to enumerate some possible directions for future research. Some of them were not treated in this thesis because of inexorable priority and timing concerns. Some others are open questions raised after the interpretation and discussion of the results achieved here.

We would like to continue this research journey, but, as said by Tolkien, others are invited to join us, because someone else has to continue this story...

In this way, without any intention to be complete, some questions can be listed:

- how to take advantage of sensitivity analysis methods for better variable and design space definition and enhance the optimization algorithms behavior and results?
- how to incorporate or integrate EGO algorithms to handle reliability engineering and robust optimization?
- how to take advantage of experimental data allied to computer results and function gradients, to enhance the metamodel prediction capabilities?;
- is it possible to treat inverse problems by using EGO-family algorithms?
- is it possible to smooth the expected improvement functions and improve the EGO convergence?

- what to expect from the application of the EGO methods to other classes of problems such as: multiboby dynamics, noise vibration and harshness (NVH) and other ill-posed and noisy problems like optimization of eigen-problems?

As we already discussed, other fronts of future research can be followed, for instance, the treatment of multiobjective optimization problems and better treatment of constraints to avoid unfeasible or inactive results.

Also, the correct handling of discrete and stochastic variables for robust optimization is extremely important when the design variables and functions are not continuous and non-deterministic.

In addition, objective and systematic convergence criteria is still a demand for metamodel based optimization and EGO-type algorithms as well. In the same way, methods for reducing the number of sampling points (true function evaluations) and, at same time, accelerating the overall convergence need to be investigated.

In this sense, the derivation of hybrid methods by combining classical sequential sampling or other adaptive approaches (clustering, etc.) with EGO-type algorithms can be a promising front of research.

These and other possible extensions of EGO-type algorithms are of interest for research and practical applications and we intend to explore this fields in our future work.

References

Acar E (2010) Various approaches for constructing an ensemble of metamodels using local error measures. Structural and Multidisciplinary Optimization 42(6):879–896

Acar E, Rais-Rohani M (2009) Ensemble of metamodels with optimized weight factors. Structural and Multidisciplinary Optimization 37(3):279–294

Akaike H (1974) A new look at the statistical model identification. IEEE Transactions on Automation and Control 19:716–723

Alghamdi AAA (2001) Collapsible impact energy absorbers: an overview. Thin-Walled Structures 39:189–213

Ali MM, Khompatraporn C, Zabinski Z (2005) A numerical evaluation of several stochastic algorithm on selected continuous global optimization test problems. Journal of Global Optimization 31:635–672

Amemiya T (1985) Advanced Econometrics. Harvard University Pres, Cambridge, Massachsetts, USA

Arora JS (1976) Survey of structural reanalysis techniques. Journal of Strucutral Division - ASCE 102:783–802

Arora JS (2012) Introduction to Optimum Design, 3^{rd} edn. Elsevier

Barthelemy JFM, Haftka RT (1993) Approximation concepts for optimum design - a review. Structural Optimization 5:129–144

Bishop CM (1995) Neural Networks for Pattern Recognition. Oxford University Press Inc., New York, USA

Björk A (1996) Numerical Methods for Least Squares Problems. SIAM: Society for Industrial and Applied Mathematics

Box GEP (1954) The exploration and exploitation of response surfaces: Some general considerations and examples. Biometrics 10:16–60

Box GEP, Draper NR (1987) Empirical model building and response surfaces. John Wiley and Sons, New York, USA

Box GEP, Hunter WG, Hunter SJ (1978) Statistics for Experimenters: An Introduction to Design, Data Analysis, and Model Building. John Wiley and Sons

Breiman L (1996) Stacked regressions. Machine Learning 24:49–64

Buhmann MD (2003) Radial Basis Functions: Theory and Implementations. Cambridge Monographs on Applied and Computational Mathematics, Cambridge University Press, Cambridge, UK

Carpenter WC, Barthelemy JFM (1992) Comparison of polynomial approximations and artificial neural nets for response surfaces in engineering optimization. In: 33rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference, Dallas-TX-USA

Chaudhuri A, Haftka RT (2012) Efficient global optimization with adaptive target setting. AIAA Journal 52(7):1573–1578

Coello CAC (2000) Use of a self-adaptive penalty approach for engineering optimization problems. Computers in Industry 41(2):113–127

Desautels T, Krause A, Burdick J (2012) Parallelizing exploration-exploitation tradeoffs in gaussian process bandit optimization. The Journal of Machine Learning Research 15(1):13,873–3923

Efroymson MA (1960) Multiple regression analysis. In: Mathematical Methods for Digital Computers, Wiley, New York, USA, pp 191–203

Elmarakbi A, Long YX, McIntyre J (2013) Crash analysis and energy absorption fo s-shapped longitudinal memebers. Thin-Walled Structures 68:65–74

Encisoa SM, Branke J (2015) Tracking global optima in dynamic environments with efficient global optimization. European Journal of Operational Research 242:744–755

Fan J, Li R (2001) Variable selection via nonconcave penalized likelihood and its oracle properties. Journal of American Statistical Association 96(456):1348–1360

Fang H, Rais-Rohani M, Liu Z, Horstemeyer MF (2005) A comparative study of metamodeling mehtods for multiobjective crashworthiness optimization. Computers & Structures 83:2121–2136

Fang KT, Li R, Sudjianto A (2006) Design and Modeling for Computer Experiments. Computer Science and Data Analysis Series, Chapman & Hall/CRC, Boca Raton, USA

Fasshauer GE (2007) Meshfree Approximation Methods with MATLAB, Interdisciplinary Mathematical Sciences, vol 6. World Scientific, Singapore

Ferreira WG, Serpa AL (2016) Ensemble of metamodels: The augmented least squares approach. Structural and Multidisciplinary Optimization 53(5):1019–1046

Ferreira WG, Alves P, Slave R, Attrot W, Magalhaes M (2012) Optimization of a CLU truck frame. In: Ford Global Noise & Vibration Conference, Ford Motor Company, PUB-NVH108-02

Fierro RD, Bunch JR (1994) Collinearity and total least squares. SIAM Journal of Matrix Analysis Applications 15:1167–1181

Fierro RD, Bunch JR (1997) Regularization by truncated total least squares. SIAM Journal of Scientific Computation 18(4):1223–1241

Forrester A, Keane A (2009) Recent advances in surrogate-based optimization. Progress in Aerospace Sciences 45:50–79

Forrester A, Sóbester A, Keane A (2008) Engineering Desing Via Surrogate Modelling - A Practical Guide. John Wiley & Sons, United Kingdom

Foster DP, George EI (1994) The risk inflation criterion for multiple regression. Annals of Statistics 22:1947–1975

Frank IE, Friedman JH (1993) A statistical view on some chemometrics regression tools. Technometrics 35:109–148

Garthwaite PH (1994) An interpretation of partial least squares. Journal of the American Statistical Association 8(425):122–127

Ge Q, Ciuffo B, Menendez M (2015) Combining screening and metamodel-based methods: An efficient sequential approach for the sensitivity analysis of model outputs. Reliability Engineering and System Safety 134:334344

Ginsbourger D, Riche RL, Carraro L (2010) Kriging is well-suited to parallelize optimization. In: Computational Intelligence in Expensive Optimization Problems - Adaptation Learning and Optimization, Springer, vol 2, pp 131–162

Giunta AA, Watson LT (1998) Comparison of approximation modeling techniques: polynomial versus interpolating models. In: 7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, AIAA-98-4758, pp 392–404

Goel T, Haftka RT, Shyy W, Queipo NV (2007) Ensemble of surrogates. Structural and Multidisciplinary Optimization 33:199–216

Golub GH, van Loan CH (1980) An analysis of total least squares problem. SIAM Journal of Numerical Analysis 17:883–893

Golub GH, Heath M, Wahba G (1979) Generalizaed cross-validation as a method for choosing a good ridge parameter. Technometrics 21(2):215–223

Gunn SR (1997) Support vector machines for classification and regression. Technical Report. Image, Speech and Inteligent Systems Research Group. University of Southhampton, UK

Haftka RT, Scott EP, Cruz JR (1998) Optimization and experiments: a survey. Applied Mechanical Reviews 7:435–448

Haftka RT, Villanueva D, Chaudhuri A (2016) Parallel surrogate-assisted global optimization with expensive functions - a survey. Structural and Multidisciplinary Optimization 54(1):3–13

Hajela P, Berke L (1990) Neurobiological computational models in structural analysis and design. In: 31st AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference, Part I, Long Beach-CA-USA, pp 345–355

Han ZH, Zhang KS (2012) Surrogate-Based Optimization - Real-World Application of Genetic Algorithms, ISBN 978-953-51-0146-8 edn. InTech, Dr. Olympia Roeva - Editor, Shanghai, China

Hannan EJ, Quinn BG (1979) The determination of the order of autoregression. Journal of Royal Statistics Society - Series B 41:190–195

Hardy RL (1971) Multiquadric equations of topography and other irregular surfaces. Journal of Geophysical Research 76:1905–1915

Hashem S (1993) Optimal linear combinations of neural networks. PhD thesis, School of Industrial Engineering. Purdue University, West Lafayette, IN, USA

Henkenjohann N, Kukert J (2007) An efficient sequential optimization approach based on the multivariate expected improvement criterion. Quality Engineering 19(4):267–280

Hoerl AE, Kennard RW (1970a) Ridge regression: Applications to nonorthogonal problems. Technometrics 12(1):69–82

Hoerl AE, Kennard RW (1970b) Ridge regression: Biased estimation for nonorthogonal problems. Technometrics 12(1):55–67

Hoerl AE, Kennard RW (1976) Ridge regression: Iterative estimation of the biasing parameter. Communications in Statistics A5:77–88

Hoerl AE, Kennard RW, Baldwin KF (1975) Ridge regression: Some simulations. Communications in Statistics 4:105–123

Huber PJ, Rochetti EM (2009) Robust Statistics. Wiley Series in Probability and Statistics, John Wiley & Sons, Hoboken, New Jersey

van Huffel S, Vandewalle J (1987) Classical regression and total least-squares estimation. Linear Algebra and its Applications 93:149–160

van Huffel S, Vandewalle J (1991) The Total Least Squares Problem: Computational Aspects and Analysis. SIAM, Philadelphia, USA

Hunter DR, Li R (2005) Variable selection using mm algorithms. Annals of Statistics 33:1617–1642

Ioss B, Lemaitre P (2015) A review on global sensitivity analysis methods. In: Uncertainty management in simulation-optimization of complex systems: algorithms and applications, Springer, pp 1–23

Janusevskis J, Riche RL, Ginsbourger D, Girdziusas R (2012) Expected improvements for the asynchronous parallel global optimization of expensive functions: Potentials and challenges. Learning and Intelligent Optimization 7219:413–418

Jekabsons G (2009) RBF: Radial basis function interpolation for matlab/octave. Riga Technical University, Latvia, version 1.1 ed.

Jin R, Chen W, Sudjianto A (2002) On sequential sampling for global metamodeling in engineering design. In: Engineering Technical Conferences and Computers and Information in Engineering Conference, DETC2002/DAC-34092, ASME 2002 Design, Montreal-Canada

Jolliffe IT (2002) Principal Component Analysis. Springer Series in Statistics, Springer, New York, USA

Jones DR (2001) A taxonomy of global optimization methods based on response surfaces. Journal of Global Optimization 21:345–383

Jones DR, Schonlau M, Welch WJ (1998) Efficient global optimization of expensive black-box functions. Journal of Global Optimization 13:455–492

Jurecka F (2007) Optimization based on metamodeling techniques. PhD thesis, Technische Universität München, München-Germany

Kannan BK, Kramer SN (1994) An augmented lagrange multiplier based method for mixed integer discrete continuous optimization and its applications to mechanical design. Journal of Mechanical Design 116:318–320

Kleijnen JPC (1985) Statistical Tools for Simulation Practioners. Marcel Decker, New York

Kleijnen JPC (2009) Kriging metamodeling in simulation: A review. European Journal of Operational Research 192(3):707–716

Koziel S, Leifesson L (2013) Surrogate-Based Modeling and Optimization - Applications in Engineering. Springer, New York, USA

Krige DG (1951) A statistical approach to some mine valuations and allied problems at the witwatersrand. Master's thesis, University of Witwatersrand, Witwatersrand

Kuan CM (2012) Econometrics - Lecture Notes. Available at: `http:homepage.ntu.edu.tw/c̃kuan/e-about.html`. National Taiwan University, Taipei, Taiwan

Lai KK, Yu L, Wang SY, , Wei H (2006) A novel nonlinear neural network ensemble forecasting model for financial time series forecasting. In: Lecture Notes in Computer Science 3991, pp 790–793

Lemonge AC, Barbosa HJ, Borges CC, Silva FB (2010) Constrained optimization problems in mechanical engineering design using a real-coded steady-state genetic algorithm. In: Dvorkin E, Goldschmit M, Storti M (eds) Mecánica Computacional Vol. XXIX, Associación Argentina de Mecánica Computacional, Buenos Aires, pp 9287–9303

Lophaven SN, Nielsen HB, Sondergaard J (2002) DACE - a matlab kriging toolbox. Tech. Rep. IMM-TR-2002-12, Technical University of Denmark

Madu CN, Kuei CH (1994) Regression metamodeling in computer simulation - the state of the art. Simulation Practice and Theory 2:27–41

Markovsky I, van Huffel S (2007) Overview of total least-squares methods. Signal Processing 87:2283–2302

Meckesheimer M, et al (2002) Computationally inexpensive metamodel assessment strategies. AIAA Journal 40(10):2053–2059

Mehari MT, Poorter E, Couckuyt I, Deschrijver D, Gerwen JV, Pareit D, Dhaene T, Moerman I (2015) Efficient global optimization of multi-parameter network problems on wireless testbeds. Ad Hoc Networks 29:15–31

Meng C, Wu J (2012) A novel nonlinear neural network ensemble model using k-plsr for rainfall forecasting. In: Bio-Inspired Computing Applications. Lecture Notes in Computer Science 6840, pp 41–48

Midi H, Hua LU (2009) The performance of latent root-m based regression. Journal of Mathematics and Statistics 5(1):1–9

Miller A (2002) Subset Selection in Regression. Monographs on Statistics and Applied Probability, Chapman & Hall/CRC, USA

Mockus J (1994) Application of bayesian approach to numerical methods of global and stochastic optimization. Journal of Global Optimization 4:347–365

Montgomery DC, Peck EA, Vining GG (2006) Introduction to Linear Regression Analysis. Wiley Series in Probability and Statistics, John Wiley & Sons, Hoboken, New Jersey

Morris MD (1991) Factorial sampling plans for preliminary computer experiments. Technometrics 33(2):161–174

Myers RH, Montgomery DC (2002) Response Surface Methodology: Process and Product Optimization Using Designed Experiments, 2nd edn. John Wiley and Sons, New York

Ng S (2012) Variable selection in predictive regressions. In: Handbook of Economical Forecasting, Elsevier, pp 752–789

Papalambros PY (2002) The optimization paradigm in engineering design: promises and challenges. Computer-Aided Design 34:939–951

Perrone MP, Cooper LN (1993) When networks disagree: Ensemble methods for hybrid neural networks. Artificial Neural Networks for Speech and Vision, Chapman & Hall, London, UK

Ponweiser W, Wagner T, Vincze M (2008) Clustered multiple generalized expected improvement: A novel infill sampling criterion for surrogate models. In: Wang J (ed) 2008 IEEE World Congress on Computational Intelligence, IEEE Computational Intelligence Society, IEEE Press, Hong Kong, pp 3514–3521

Queipo NV, Pintos S, Nava E (2013) Setting targets for surrogate-based optimization. Journal of Global Optimization 55(4):857–875

Queipo NV, et al (2005) Surrogate-based analysis and optimization. Progress in Aerospace Sciences 41:1–28

Ramu M, Prabhu RV (2013) Metamodel based analysis and its applications: A review. Acta Technica Corviniensis - Bulletin of Engineering 4(2):25–34

Rao SS (2009) Engineering Optimization: Theory and Practice. John Wiley & Sons, Inc., 4th Edition.

Rasmussen CE, Williams CK (2006) Gaussian Processes for Machine Learning. The MIT Press

Rehman SU, Langelaar M, Keulen FV (2014) Efficient kriging-based robust optimization of unconstrained problems. Journal of Computational Science 5:872–881

Rousseeuw PJ, Leroy AM (2003) Robust Regression and Outlier Detection. Wiley Series in Probability and Statistics, John Wiley & Sons, Hoboken, New Jersey

Roux WJ, Stander N, Haftka RT (1998) Response surface approximations for structural optimization. International Journal for Numerical Methods in Engineering 42:517–534

Roy R, et al (2008) Recent advances in engineering design optimisation: challenges and future trends. CIRP Annals - Manufacturing Technology 57:697–715

Sacks J, Schiller J, Welch W (1989a) Designs for computer experiments. Technometrics 31:41–47

Sacks J, Welch W, Michell TJ, Wynn HP (1989b) Design and analysis of computer experiments. Statistical Science 36:409–435

Saitou K (2005) A survey of structural optimization in mechanical product development. Transactions of the ASME 65:214–226

Saltelli A, Tarantola S, Campolongo F, Ratto M (2004) Sensitivity Analysis in Practice: A Guide to Assessing Scientific Models. John Wiley & Sons, West Sussex, England

Sanchez E, Pintos S, Queipo NV (2008) Toward and optimal ensemble of kernel-based approximations with engineering applications. Structural and Multidisciplinary Optimization 36:247–261

Santner TJ, Williams BJ, Notz WI (2003) The Design and Analysis of Computer Experiments. Springer Series in Statistics, Springer-Verlag, New York, USA

Scheipl F, Kneib T, Fahrmeir L (2013) Penalized likelihood and bayesian function selection in regression models. Advances in Statistical Analysis 97(4):349–385

Schmit LA, Farshi B (1974) Some approximation concepts for structural synthesis. AIAA Journal 12:692–699

Schmit LA, Miura H (1976) Approximation concepts for efficient strutural synthesis. NASA CR-2552

Schonlau M (1997) Computer experiments and global optimization. PhD thesis, University of Waterloo, Watterloo, Ontario, Canada

Schwarz G (1978) Estimating the dimension of a model. Annals of Statistics 6:461–464

Seni G, Elder J (2010) Ensemble Methods in Data Mining: Improving Accuracy Through Combining Predictions. Synthesis Lectures on Data Mining and Knowledge Discovery, Morgan & Claypool Publishers, Chicago, IL, USA

Shibata R (1984) Approximation efficiency of a selection procedure for a number of regression variables. Biometrika 71:43–49

Simpson TW, Peplinski J, Koch PN, Allen JK (2001a) Metamodels for computer-based engineering design: Survey and recommendations. Engineering with Computers 17(2):129–150

Simpson TW, Toropov V, Balabanov V, Viana FAC (2008) Design and analysis of computer experiments in multidisciplinary design optimization: A review of how far we have come - or not. In: 12th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Victoria, British Columbia

Simpson TW, et al (2001b) Kriging models for global approximation in simulation-based multidisciplinary design optimization. AIAA Journal 39(12):2233–2241

Smola AJ, Schölkopf B (1998) A tutorial on support vector regression. NeuroCOLT2 Technical Report Series. NC2-TR-1998-030, Berlin, Germany

Sóbester A, Leary SJ, Keane A (2004) A parallel updating scheme for approximating and optimizing high fidelity computer simulations. Structural and Multidisciplinary Optimization 27:371–383

Sóbester A, Leary SJ, Keane A (2005) On the design of optimization strategies based on global response surface approximation models. Journal of Global Optimization 33:31–59

Sobieszczanski-Sobieski J, Haftka RT (1997) Multidisciplinary aerospace design optimization: Survey and recent developments. Structural Optimization 14:1–23

Tanlak N, Sonmez FO (2014) Optimal shape design of thin-walled tubes under high velocity axial impact loads. Thin-Walled Structures 84:302–312

Thacker WI, Zhang J, Watson LT, Birch JB, Iyer MA, Berry MW (2010) Algorithm 905: SHEPPACK: modified shepard algorithm for interpolation of scattered multivariate data. ACM Transactions on Mathematical Software 37(3):1–20

Tibshirani R (1996) Regression shrinkage and selection via lasso. Journal of Royal Statistical Society 58(1):267–288

Vapnik V, Lerner A (1963) Patter recognition using generalized portrait method. Autom Remote Control (English Translation) 24(6):774–780

Vapnik VN (2000) The Nature of Statistical Learning Theory, 2nd edn. Springer-Verlag

Venkatararaman S, Haftka RT (2004) Structural optimization complexity: what has moore's law done for us? Structural and Multidisciplinary Optimization 28:375–287

Viana FAC (2009) SURROGATES toolbox user's guide version 2.0 (release 3). Available at website: `http://fchegury.googlepages.com`

Viana FAC (2011) Multiples surrogates for prediction and optimization. PhD thesis, University of Florida, Gainesville, FL, USA

Viana FAC, Haftka RT (2010) Surrogage-based optimization with parallel simulations using probability of improvement. In: Proceedings of the 13th AIAA/SSMO Multidisciplinary Analysis Optimization Conference, Forth Worth, Texas, USA

Viana FAC, Haftka RT, Steffen V (2009) Multiple surrogates: how cross-validation error can help us to obtain the best predictor. Structural and Multidisciplinary Optimization 39(4):439–457

Viana FAC, Cogu C, Haftka RT (2010) Making the most out of surrogate models: tricks of the trade. In: Proceedings of the ASME 2010 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference IDETC/CIE 2010, Montreal, Quebec, Canada

Viana FAC, Haftka RT, Watson LT (2013) Efficient global optimization algorithm assisted by multiple surrogates techniques. Journal of Global Optimization 56:669–689

Wang GG, Shan S (2007) Review of metamodeling techniques in support of engineering design optimization. ASME Journal of Mechanical Design 129(4):370–380

Weisberg S (1985) Applied Linear Regression. Wiley Series in Probability and Statistics, John Wiley & Sons, New Jersey, USA

Wendland H (2005) Scattered Data Approximation. Cambridge Monographs on Applied and Computational Mathematics, Cambridge University Press, UK

Wolpert D (1992) Stacked generalizations. Neural Networks 5:241–259

Yang XS, Huyck C, Karamanoglu M, Khan N (2013a) True global optimality of the pressure vessel design problem: a benchmark for bio-inspired optimisation algorithms. International Journal of Bio-Inspired Computation 5(6):329335

Yang XS, Koziel S, Liefsson L (2013b) Computational optimization, modeling and simulation: Recent trends and challenges. Procedia Computer Science 18:855–860

Yu L, Wang SY, Lai KK (2005) A novel nonlinear ensemble forecasting model incorporating glar and ann for foreign exchange rates. Computers and Operations Research 32:2523–2541

Zerpa LE, Queipo NV, Pintos S, Salager JL (2005) An optimization methodology of alkaline-surfactant-polymer flooding processes using field scale numerical simulation and multiple surrogates. Journal of Petroleum Science and Engineering 47:197–208

Zhang C, Ma Y (2012) Ensemble Machine Learning. Methods and Applications. Springer, New York, USA

Zhang X, Zhang H, Wen Z (2015) Axial crushing of tapered circular tubes with graded thickness. International Journal for Mechanical Sciences 92:12–23

Zhang Y, Sun G, Xu X, Li G, Li Q (2014) Multiobjective crashworthiness optimization of howllow and conical tubes for multiple load cases. Thin-Walled Structures 82:331–342

Zhou ZH (2012) Ensemble Methods. Foundations and Algorithms. Machine Learning & Pattern Recognition Series, Chapman & Hall/CRC, Boca Raton, USA

A SURROGATES Toolbox

The SURROGATES Toolbox (ref. Viana (2009)) is a Matlab[14] based toolbox that aggregates and extends several open-source tools previously developed in the literature for design and analysis of computer experiments, i.e., metamodeling and optimization. We used the version v2.0, but v3.0 already includes EGO variants[15].

The SURROGATES Toolbox uses the following collection of third party software published: SVM by Gunn (1997), DACE by Lophaven et al (2002), GPML by Rasmussen and Williams (2006), RBF by Jekabsons (2009), and SHEPPACK by Thacker et al (2010). The compilation in a single framework has been implemented and applied in previous research by Viana and co-workers, as for example Viana et al (2009) and Viana (2011).

[14]Matlab is a well known and widely used numerical programing platform and it is developed and distributed by The Mathworks Inc., see `www.mathworks.com`.

[15]Further details and recent updates of SURROGATES Toolbox refer to the website: `https://sites.google.com/site/srgtstoolbox/`.

B Boxplots Definition

Boxplot is a common statistical graph used for visual comparison of the distribution of different stochastic variables in a same plane. We used the Matlab function `boxplot` (with default parameters) to create the boxplots in the present work. See in Fig.54 an example of boxplot, with the distribution of four random variables.

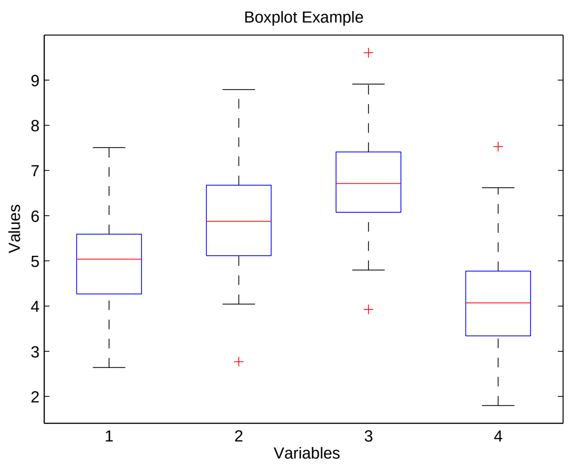


Figure 54: Boxplot example with the distribution of four random variables. Each box is defined by lines at the lower quartile (25%), median (50%) and upper quartile (75%) of the data. Lines extending above and upper each box (*whiskers*) indicate the spread for the rest of the data out of the quartiles definition. If existent, outliers are represented by plus signs “+”, above/below the whiskers.

C The Kriging Metamodel

Kriging model, originally proposed by Krige (1951), is an interpolating metamodel in which the basis functions, as stated in Eq. 1, are of the form

$$\psi^{(i)} = \psi\left(\left\|\mathbf{x}^{(i)} - \mathbf{x}\right\|\right) = \exp\left(-\sum_{j=1}^{k}\theta_j \left|x^{(i)} - x_j\right|^{p_j}\right), \tag{33}$$

with tuning parameters θ_j and p_j normally determined by maximum likelihood estimates.

With the parameters estimated, the final kriging predictor is of the form

$$\hat{f}(\mathbf{x}) = \hat{\mu} + \boldsymbol{\psi}^T \boldsymbol{\Psi}^{-1}\left(\mathbf{y} - \mathbf{1}\hat{\mu}\right), \tag{34}$$

where $\mathbf{y} = \left[y^{(1)} \ldots y^{(N)}\right]^T$, $\mathbf{1}$ is a vector of ones, $\boldsymbol{\Psi} = \psi^{(r)(s)}$ is the so called $N \times N$ *matrix of correlations* between the sample data, calculated by means of Eq. 33 as

$$\boldsymbol{\Psi} = \psi\left(\left\|\mathbf{x}^{(r)} - \mathbf{x}^{(s)}\right\|\right) \tag{35}$$

and $\hat{\mu}$ is given by

$$\hat{\mu} = \frac{\mathbf{1}^T \boldsymbol{\Psi}^{-1} \mathbf{y}}{\mathbf{1}^T \boldsymbol{\Psi}^{-1} \mathbf{1}}. \tag{36}$$

One of the key benefits of kriging models is the provision of uncertainty estimate for the prediction (mean squared error, MSE) at each point $\mathbf{x}$, given by

$$\hat{s}^2(\mathbf{x}) = \hat{\sigma}^2\left[1 - \boldsymbol{\psi}^T \boldsymbol{\Psi}^{-1}\boldsymbol{\psi} + \frac{1 - \mathbf{1}^T \boldsymbol{\Psi}^{-1}\mathbf{y}}{\mathbf{1}^T \boldsymbol{\Psi}^{-1}\mathbf{1}}\right], \tag{37}$$

with variance estimated by

$$\hat{\sigma}^2 = \frac{\left(\mathbf{y} - \mathbf{1}\hat{\mu}\right)^T \boldsymbol{\Psi}^{-1}\left(\mathbf{y} - \mathbf{1}\hat{\mu}\right)}{N}. \tag{38}$$

Refer to Forrester et al (2008) or Fang et al (2006) for further details on metamodel formulation.

D Test Functions

D.1 Analytical Benchmarks

In the reference Ali et al (2005) it is presented as appendix “A Collection Benchmark Global Optimization Test Problems”, that compiles 50 test functions from the optimization related literature. These kind of functions are widely used to validate both metamodeling and optimization methods, as for example Goel et al (2007), Acar and Rais-Rohani (2009), Acar (2010), Viana et al (2009) and Viana et al (2013). The functions used and referenced along this thesis are the following.

Branin-Hoo

$$y(\mathbf{x}) = \left(x_2 + \frac{5.1x_1^2}{4\pi^2} + \frac{5x_1}{\pi} - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos(x_1) + 10, \tag{39}$$

for the region $-5 \leq x_1 \leq 10$ and $0 \leq x_2 \leq 15$. There are 3 minimum points in this region, i.e., $\mathbf{x}^* \approx (-\pi, 12.275), (\pi, 2.275), (3\pi, 2.475)$ with $f(\mathbf{x}^*) = \frac{5}{4\pi}$.

Hartman

$$y(\mathbf{x}) = -\sum_{i=1}^{4} c_i \exp\left[-\sum_{j=1}^{n_v} a_{ij}\left(x_j - p_{ij}\right)^2\right], \tag{40}$$

where $x_i \in [0,1]^{n_v}$, with constants c_i, a_{ij} and p_{ij} given in Table 3, for the case $n_v = 3$ (Hartman-3); and in Table 4 and Table 5, for the case $n_v = 6$ (Hartman-6).

In case of Hartman-3, there are four local minima,

$$\mathbf{x}_{local} \approx (p_{i1}, p_{i2}, p_{i3}),$$

with $f_{local} \approx -c_i$ and the global minimum is located at

$$\mathbf{x}^* \approx (0.114614, 0.555649, 0.852547),$$

with $f(\mathbf{x}^*) \approx -3.862782$.

In case of Hartman-6, there are four local minima,

$$\mathbf{x}_{local} \approx (p_{i1}, p_{i2}, p_{i3}, p_{i4}, p_{i5}, p_{i6}),$$

with $f_{local} \approx -c_i$ and the global minimum is located at

$$\mathbf{x}^* \approx (0.201690, 0.150011, 0.476874, 0.275332, 0.3111652, 0.657301),$$

with $f(\mathbf{x}^*) \approx -3.322368$.

Table 3: Data for Hartman-3 function, c_i and a_{ij} and p_{ij} .

i	c_i	a_{ij}			p_{ij}		
		$j=1$	2	3	$j=1$	2	3
1	1	3	10	30	0.3689	0.117	0.2673
2	1.2	0.1	10	35	0.4699	0.4387	0.747
3	3	3	10	30	0.1091	0.8732	0.5547
4	3.2	0.1	10	35	0.03815	0.5743	0.8828

Table 4: Data for Hartman-6 function, c_i and a_{ij}.

i	c_i	a_{ij}					
		$j=1$	2	3	4	5	6
1	1	10	3	1	3.5	1.7	8
2	1.2	0.05	10	17	0.1	8	14
3	3	3	3.5	1.7	10	17	8
4	3.2	17	8	0.05	10	0.1	14

Table 5: Data for Hartman-6 function, p_{ij}.

i	p_{ij}					
	$j=1$	2	3	4	5	6
1	0.1312	0.1696	0.5569	0.0124	0.8283	0.5886
2	0.2329	0.4135	0.8307	0.3736	0.1004	0.9991
3	0.2348	0.1451	0.3522	0.2883	0.3047	0.665
4	0.4047	0.8828	0.8732	0.5743	0.1091	0.0381

Giunta-Watson This is the "noise-free" version of the function used by Giunta and Watson (1998)

$$y(\mathbf{x}) = \sum_{i=1}^{n_v} \left[\frac{3}{10} + \sin\left(\frac{16}{15}x_i - 1\right) + \sin^2\left(\frac{16}{15}x_i - 1\right) \right], \tag{41}$$

where $\mathbf{x} \in [-2, 4]^{n_v}$. For this domain and one variable, $i = 1$, the local maximum is located at

$$x^* \approx 2.74218,$$

with $f(x^*) \approx 2.29999$. In case of two variables, $i = 2$, the local maximum is located at

$$\mathbf{x}^* \approx (2.74218, 2.74218)$$

with $f(\mathbf{x}^*) \approx 4.59999$.

Aircraft Wing Weight (*LiftSurf*) This is a function used in Forrester et al (2008), pp.10-13. It is based on Raymer (2006) and represents roughly the weight of a Cessna C172 Skyhawk aircraft for preliminary design studies. The 10 variable version for the function is

$$W = 0.036 S_w^{0.758} W_{fw}^{0.0035} \left(\frac{A}{\cos^2 \Lambda}\right)^{0.6} q^{0.006} \lambda^{0.04} \left(\frac{100 t_c}{\cos \Lambda}\right)^{-0.3} (N_z W_{dg})^{0.49} + S_w W_p \ , \qquad (42)$$

where the parameters (design variables) are described in the Table 6. After sensitivities studies it was identified that N_z, W_{dg}, S_w and t_c are the most important variables for this problem. Then, by keeping the remaining variables constant in their baseline values, the problem can reduced to 4 variables.

Table 6: Nomenclature for the aircraft wing weight model.

Symbol	Parameter	Baseline	Lower Bound	Upper Bound
S_w	Wing area (ft^2)	174	150	200
W_{fw}	Weight of the fuel in the wing (lbf)	252	220	300
A	Aspect ratio	7.52	6	10
Λ	Quarter-chord sweep (deg)	0	-10	10
q	Dynamic pressure at cruise (lbf/ft^2)	34	16	45
λ	Taper ratio	0.672	0.5	1
t_c	Aerofoil thickness to chord ratio	0.12	0.08	0.18
N_z	Ultimate load factor	3.8	2.5	6
W_{dg}	Flight design gross weight (lbf)	2000	1700	2500
W_p	Paint weight (lbf/ft^2)	0.064	0.025	0.08

D.2 Engineering Applications

In Figs. 55 and 56 are presented simulation models currently applied in automotive industry. These models are typical examples of the ones used in the Multidisciplinary Optimization (MDO) department at Ford Motor Company, where the author of the present thesis works as structural optimization engineer. The examples described in this section are taken only as illustrations and they were part of a MDO project presented in a restrict conference summarized in the report by Ferreira et al (2012).

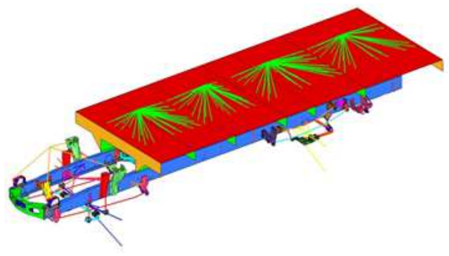
A regular MDO study at early design phases can comprise several models with hundreds of design variables and response functions to be monitored. After design sensitivity analysis stage, the top most significant variables and functions are selected in each model for metamodeling and multidisciplinary optimization.

We will use in this work the data available for the following variables and responses regarding these models to compare the performance of the ensemble methods discussed in this work by means of real-world applications.

Truck Models and Responses

a) Truck Durability: it is presented in Fig. 55(a) a finite elements (FEM) model build in NASTRAN for truck frame durability evaluation. The durability responses (i.e., stress and/or fatigue/endurance metrics) are described as function of $n_v = 12$ geometry variables;

b) Truck Dynamics: it is presented in Fig. 55(b) a multibody model build in ADAMS for vehicle dynamics evaluation. The dynamics responses (i.e., displacements, velocities or accelerations for ride and handling performance) are defined based on the same $n_v = 12$ geometry variables used in the durability responses.


(a) Durability

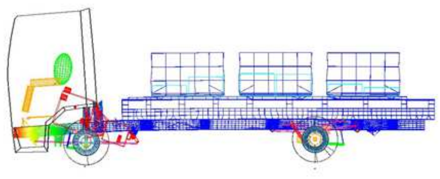

(b) Vehicle Dynamics

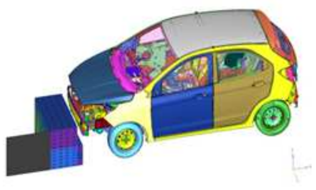
Figure 55: Examples of truck models applied in automotive industry for metamodeling and optimization. Courtesy of Ford Motor Company.

Car Models and Responses

a) Car NVH: it is presented in Fig. 56(a) a FEM model build in NASTRAN for passenger car NVH (*noise, vibration and harshness*) evaluation. The NVH response is described as function of $n_v = 30$ geometry variables;

b) Car Crash: it is presented in Fig. 56(b) a FEM model in RADIOSS for passenger car Frontal Crash evaluation. The crash responses (i.e., displacements, velocities or accelerations for safety performance) are described with $n_v = 44$ variables, that is the same 30 geometry used for NVH and additional 14 material parameters.

(a) NVH



(b) Frontal Crash

Figure 56: Examples of passenger car models applied in automotive industry for metamodeling and optimization. Courtesy of Ford Motor Company.

E Preliminary Numerical Study

E.1 Introductory Note

Our preliminary studies and numerical experiments with LS ensembles, which motivated the development of this thesis, started in 2009. Since these results were not published before, we will record them here in this appendix for convenience.

E.2 Abstract

The use of approximate mathematical models as surrogates for expensive computational simulation models has became a common practice in engineering design analysis and optimization. In the last five years, some research has been conducted in order to investigate the effectiveness of selecting and combining different surrogate models. These studies suggested that a weighted averaged surrogate model (WAS) has better accuracy than individual surrogates. Most of the previous proposed WAS schemes was based on the prediction sum of squares (PRESS) and demonstrated that using an ensemble of surrogate models can improve robustness of the predictions by reducing the impact of a poorly fitted surrogate model. On the other hand, the computational cost involved in calculating an error measure such as PRESS is high, which can limit its applicability. In this work it is implemented an alternative approach for estimating weights for ensemble of surrogates models, based on the concept of the standard least squares approximation, without explicitly calculating any error measure. The numerical results of a comparative study show that the proposed method presents accuracy in the same level of magnitude as compared to the methods published in the literature. On the other hand, the computational cost of the proposed method is at least one order of magnitude faster than the others based on PRESS.

E.3 Numerical experiments

The main objective of the present work is to implement and to compare the performance of the proposed least squares (LSQ) ensemble of surrogates method against previously developed approaches based on PRESS error.

In order to complete this objective, we separate the numerical experiments in two parts. At first, we study and compare the overall behavior of LSQ in terms of accuracy, relative computational cost and error variance, as function of the number of sampling points and number of design variables, by using analytical benchmark functions. In the second test set, the performance of LSQ method is compared with his competitors in the metamodeling of real engineering problems published in the literature.

We use the SURROGATES Toolbox developed by Viana (2009). It is a MATLAB based framework that aggregates and extends several open-source tools previously developed in the literature for design and analysis of computer experiments.

The approach adopted here to conduct the numerical experiments follows the methodology traditionally applied in the field. In particular, we follow and adapt some strategies used by Viana et al (2009) to conduct our tests.

All the examples were executed in a laptop Dell Vostro 1310 (Intel Core 2 Duo T8100 2.10GHz, 4Mb RAM), running MATLAB v2006a, on Windows Vista.

E.3.1 Tests with an analytical function

In the first test set, the analytical benchmark function presented in Giunta and Watson (1998) is used. This function is described by Eq. 41 in the Appendix D.

At first, we study the performance and convergence of the WAS techniques as the number of sampling points increases. In this set, the methods are compared for the one dimensional case ($n_v = 1$), when $N = 10$, 20, 50 and 100 points sampled in the design space $\chi = x_1 \in [-2, 4]$.

Next, we compare the performance of the WAS techniques, as the number of variables increases. The cases considered are $n_v = 1$, 2, 5 and 10 variables, in the design space $\chi = \mathbf{x} \in [-2, 4]^{n_v}$. For each case, the number of sampling points is chosen based on the rule $N = 20n_v$, in order to keep the same point density as dimension changes.

As a common practice in comparative studies of metamodeling performance, in all cases investigated, we repeat the experiments with 100 different DOE to average out the influence of random sampling points on the quality of fit. The DOE are created by using the Latin Hypercube MATLAB function `lhsdesign`, optimized with `maxmin` criterion with 1000 iterations.

The ensemble of surrogates are composed with 4 distinct models, that is: PRS, KRG, RBNN and SVR, by considering the setup described in Table 7.

All the WAS options available in SURROGATES Toolbox were considered: NPWS, PWS, OWSfull, OWSconst, OWSdiag (Eqs. 14 to 17) and BestPPRESS. In the BestPRESS option, the surrogate with lower PRESS has the weight set to $w = 1$, and the others are set to $w = 0$.

Table 7: Surrogate models setup.

Model	**Modeling technique**	**Details**
PRS	Polynomial response surface	Full quadratic model.
KRG	Kriging	Quadratic regression, exponential correlation, $\theta_0 = 10$ and $10^{-2} \leq \theta_i \leq 200$
RBNN	Radial basis neural network	$Goal = (0.05\bar{y})^2$ and $Spread = 2/5$.
SVR	Support vector regression	$C = 100max(\lvert\bar{y} + 3\sigma_y\rvert, \lvert\bar{y} - 3\sigma_y\rvert)$ and $\epsilon = \sigma_y/\sqrt{N}$

Obs.1: All other parameters not mentioned are kept with default values.
Obs.2: $\bar{y}$, σ_y and N are respectively: mean and standard deviation of y and number of sampling points.
Obs.3: No attempt has been made in order to fine tuning the surrogates shape parameters.

The LSQ method has been implemented in SURROGATES Toolbox.

In all the examples, a total of $N_{test} = 2000$ test points are considered to calculate $RMSE$, as defined in Eq. 2. The cross-validation procedure is applied with $k = 10$, when $N > 20$, to balance accuracy and computational cost for PRESS calculation. Otherwise, the leave-one-out cross-validation approach is applied.

E.3.2 Tests with engineering problems

In the second test set, four engineering problems are considered. These problems were chosen since they provide a wide range of: number of variables, sampling strategies and sampling point density and functional responses with different nonlinearity and multimodality in the design space. A brief description of each problem is given as follows. **Aerospike nozzle**: this problem has been studied by Simpson et al (2001b). The objective in this example is the optimization of the propulsion system of a space shuttle rocket engine. The problem consists of 3 geometry variables and 3 functional responses, i.e., vehicle gross liftoff weight (GLOW), thrust wall pressure and nozzle weight. An experimental design with 25 runs were used to conduct the metamodeling; **Automotive crashworthiness:** the objective of this study is the crashworthiness optimization of an automobile subjected to a frontal impact test at 56.5km/h. A computer experiment with 27 runs and 10 design variables is conducted and 3 functional responses are monitored, i.e., impact energy absorbed after 20ms and 40ms and the peak acceleration at engine top. Further details are available in Fang et al (2005); **Engine block and head joint assembly:** the objective of the design in this example is to optimize the design variables to minimize the gap lift of the assembly. A computer experiment employing 27 runs and 8 variables was conducted to optimize the design of the head gasket sealing function. Other details can be found in Fang et al (2006), p.135; **Engine noise, vibration and harshness (NVH)**: this problem deals with the optimization of NVH performance of an engine in operational conditions. The problem consists of a computer experiment with 30 runs, 17 design variables and the response considered is the oil pan radiated noise as function of engine RPM. Additional information can be found in Fang et al (2006), p.208.

The metamodels setup used for the engineering problems are in general the same of the used in the previous section (see Table 7), except when the number of sampling points versus number of variables are not enough to fit a full quadratic polynomial as for PRS and KRG. In these cases, a lower order polynomial is chosen. Except for this change, all other parameters used are the same of Table 7.

Since we have no access to the true models of these problems in order to generate new sampling points and to evaluate the $RMSE$, we separate the experimental designs in two sets: one for training and another one for model validation. In all the cases studied, we repeated the experiments with 100 different DOE, to average out the influence of random sampling points on the quality of fit. In each run we sort the design matrix rows and then we separate N points for training and the remaining N_{test} points to evaluate the $RMSE$. Table 8 present details for the resulting 8 engineering test cases studied.

Table 8: Parameters of engineering problems.

ID	Description	n_v	N	N_{test}
A1	Aerospike nozzle GLOW	3	20	5
A2	Aerospike nozzle thrust	3	20	5
A3	Aerospike nozzle weight	3	20	5
C1	Car crash energy absorbed at 20ms	10	20	7
C2	Car crash energy absorbed at 40ms	10	20	7
C3	Car crash peak acceleration at engine	10	20	7
E1	Engine block and head assembly	8	20	7
E2	Engine NVH at 6000rpm	18	20	10

E.4 Results and discussion

E.4.1 Results with the analytical function

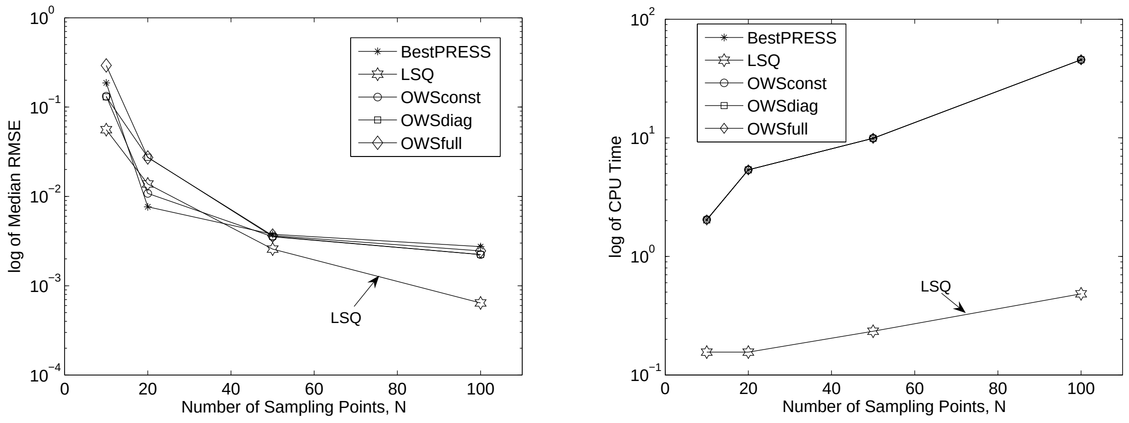
In this section the results achieved by using the analytical benchmark function (Eq. 41) will be presented.

The results of increasing the number of sampling points are presented in Table 9 and Figure 57. It is possible to observe that LSQ method presents a convergent behavior. As the number of sampling point increases, the $RMSE$ monotonically decreases. In addition the median, mean and standard deviation of $RMSE$ of LSQ (achieved over 100 runs) are in the same level of magnitude of other methods, as can be observed in Figure 57(a). On the other hand, the computational cost of LSQ is always a small fraction of other methods, i.e., 13 times faster for $N = 10$ and 94 times faster for $N = 100$. These results are illustrated in Figure 57(b).

Table 9: Effect of increasing the sampling points density.

	BestPRESS			LSQ			OWSconst			OWSdiag			OWSfull			PWS			NPWS		
N	$\bar{\bar{x}}$	$\bar{x}$	s	$\bar{\bar{x}}$	$\bar{x}$	s	$\bar{\bar{x}}$	$\bar{x}$	s	$\bar{\bar{x}}$	$\bar{x}$	s	$\bar{\bar{x}}$	$\bar{x}$	s	$\bar{\bar{x}}$	$\bar{x}$	s	$\bar{\bar{x}}$	$\bar{x}$	s
10	1.00	0.98	0.15	0.30	0.31	0.10	0.71	0.70	0.10	0.70	0.71	0.07	1.57	1.55	0.23	0.72	0.72	0.05	0.75	0.75	0.05
20	1.00	0.98	0.25	1.80	1.97	0.53	1.41	1.75	0.86	3.59	3.63	0.95	3.57	3.75	1.08	6.91	6.91	0.67	10.20	10.19	0.42
50	1.00	0.94	0.23	0.68	0.70	0.14	0.94	0.90	0.20	0.95	0.91	0.19	0.97	0.97	0.22	4.14	4.21	0.40	11.29	11.29	0.33
100	1.00	0.98	0.20	0.23	0.24	0.05	0.82	0.81	0.17	0.82	0.81	0.17	0.90	0.88	0.18	3.19	3.20	0.21	10.10	10.14	0.21

Obs.1: N is the number of sampling points, $\bar{\bar{x}}$, $\bar{x}$ and s are respectively: median, mean and standard deviation of $RMSE$ over 100 runs.
Obs.2: The values of $\bar{\bar{x}}$, $\bar{x}$ and s are normalized in relation of the median $RMSE$ measured for BestPRESS in each case of N.



(a) (b)

Figure 57: Effect of increasing the sampling points density.

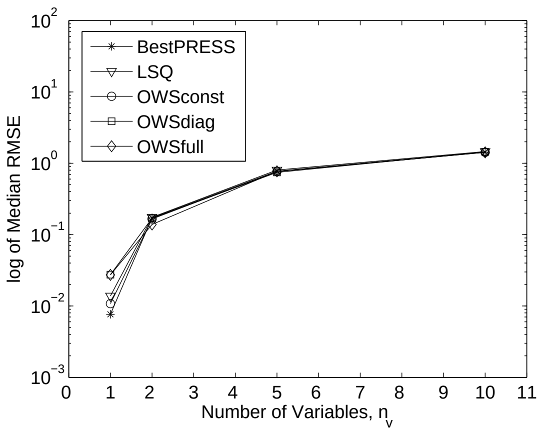
The results of increasing the number of variables are presented in Table 10 and Figure 58. Again, it is possible to observe that the median, mean and standard deviation of $RMSE$ of LSQ (achieved over 100 runs) are in the same level of magnitude of other methods, as can be seen in Figure 58(a). The same is true for the computational cost of LSQ, that is always a small fraction of other methods, i.e., 34 times faster for $n_v = 1$ and 217 times faster for $n_v = 10$. These results are illustrated in Figure 58(b).

In all cases studied, it is remarkable that more than 95% of the computational cost of the PRESS based methods is consumed in the cross-validation procedure. For instance, in the case of 10 variables, 99.98% of the CPU time was spent in PRESS calculation.

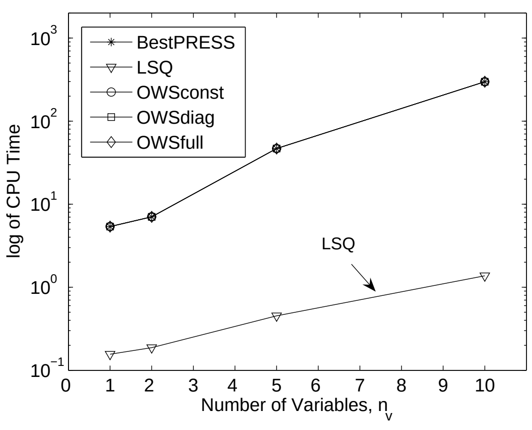
Table 10: Effect of increasing the number of variables.

	BestPRESS			LSQ			OWSconst			OWSdiag			OWSfull			PWS			NPWS		
n_v	$\bar{\bar{x}}$	$\bar{x}$	s	$\bar{\bar{x}}$	$\bar{x}$	s	$\bar{\bar{x}}$	$\bar{x}$	s	$\bar{\bar{x}}$	$\bar{x}$	s	$\bar{\bar{x}}$	$\bar{x}$	s	$\bar{\bar{x}}$	$\bar{x}$	s	$\bar{\bar{x}}$	$\bar{x}$	s
1	1.00	0.98	0.25	1.80	1.97	0.53	1.41	1.75	0.86	3.59	3.63	0.95	3.57	3.75	1.08	6.91	6.91	0.67	10.20	10.19	0.42
2	1.00	0.99	0.23	1.02	1.32	1.09	0.97	0.95	0.21	0.99	0.97	0.18	0.81	0.81	0.17	1.19	1.18	0.15	1.53	1.53	0.11
5	1.00	0.87	0.31	1.04	1.38	1.33	0.99	0.87	0.30	0.96	0.87	0.23	1.00	0.86	0.30	0.99	0.95	0.15	1.14	1.13	0.09
10	1.00	1.01	0.04	1.02	2.12	1.65	0.99	1.00	0.04	1.01	1.01	0.04	0.99	1.01	0.08	1.20	1.20	0.04	1.56	1.56	0.05

Obs.1: n_v is the number of sampling points, $\bar{\bar{x}}$, $\bar{x}$ and s are respectively: median, mean and standard deviation of $RMSE$ over 100 runs.
Obs.2: The values of $\bar{\bar{x}}$, $\bar{x}$ and s are normalized in relation of the median $RMSE$ measured for BestPRESS in each case of n_v.



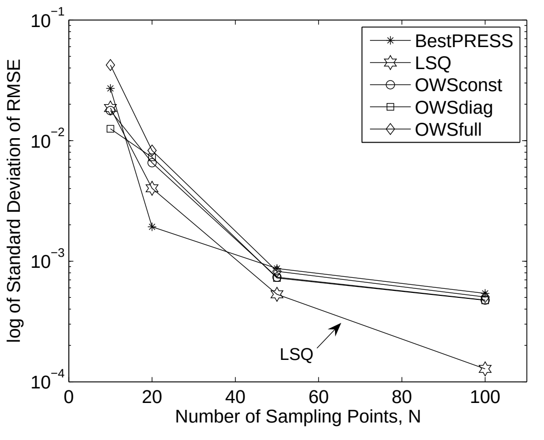
(a)



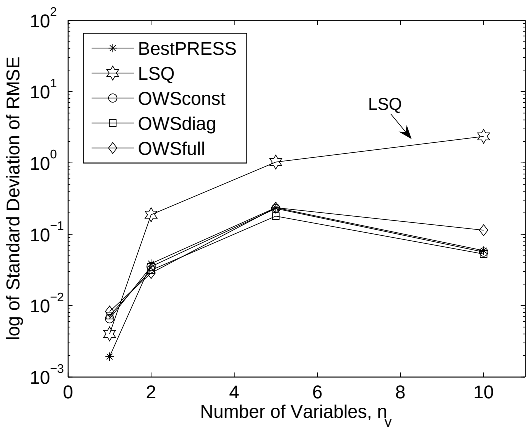
(b)

Figure 58: Effect of increasing the number of variables.

Figure 59 presents the behavior of the standard deviation of $RMSE$. It can be seen in Figure 59(a) that, as the number of sampling points increases, the variance of $RMSE$ is monotonically decreasing for all methods studied. On the other hand, the variance of LSQ method increases with the number of variables in a faster rate than the other methods, as can be seen in Figure 59(b).



(a)



(b)

Figure 59: $RMSE$ standard deviation as function of sampling points density (a) and number of variables (b).

Although the accuracy observed for LSQ is in the same level of other methods, the increasing error variance, as the model dimension increases, should cause an undesirable instability in the calculation of the WAS weights in highly nonlinear or noisy problems, specially in high dimensions. This increasing variance of LSQ may be attributed to multicollinearity of the surrogate models in the ensemble, when the sampling points distribution in the design space tends to be sparse. In other words, this effect should be attributed to a lack of linear independence (or collinearity) of the surrogates in the ensemble. As shown in Figure 59(a), this problem may be reduced by increasing the number of sampling points (with small computer cost increasing for LSQ, as previously shown in Figure 57(b)). Since the increasing of sampling points in metamodeling problems is not easy in practice, the alternative should be the implementation of some method that reduces the variance due to multicollinearity in linear regression (e.g. stepwise regression, ridge regression, principal components, etc). See for instance Miller (2002), Montgomery et al (2006) and the references therein.

In order to verify this possibility, we implemented a version of stepwise regression method for variable selection in least squares. The results of this test are presented in Figure 60. As can be observed, the stepwise variable selection method (LSQstepwise) is able to reduced the standard deviation of $RMSE$ in the calculation of the WAS weights.

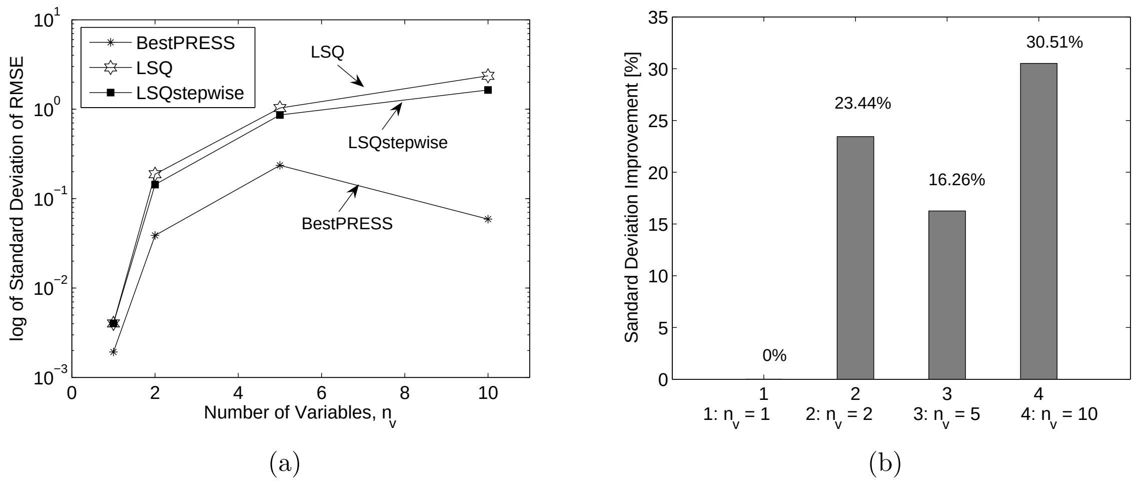


Figure 60: Improvement of $RMSE$ standard deviation based on least squares stepwise variable selection.

These preliminary results are promising and open a new front of research in this area. There are several methods available in the literature for dealing with multicollinearity in linear regression that must be investigated. In addition, it is not clear how these methods behave with different number of surrogates in the ensemble, in association with the number of sampling points and design variables.

E.4.2 Results with engineering problems

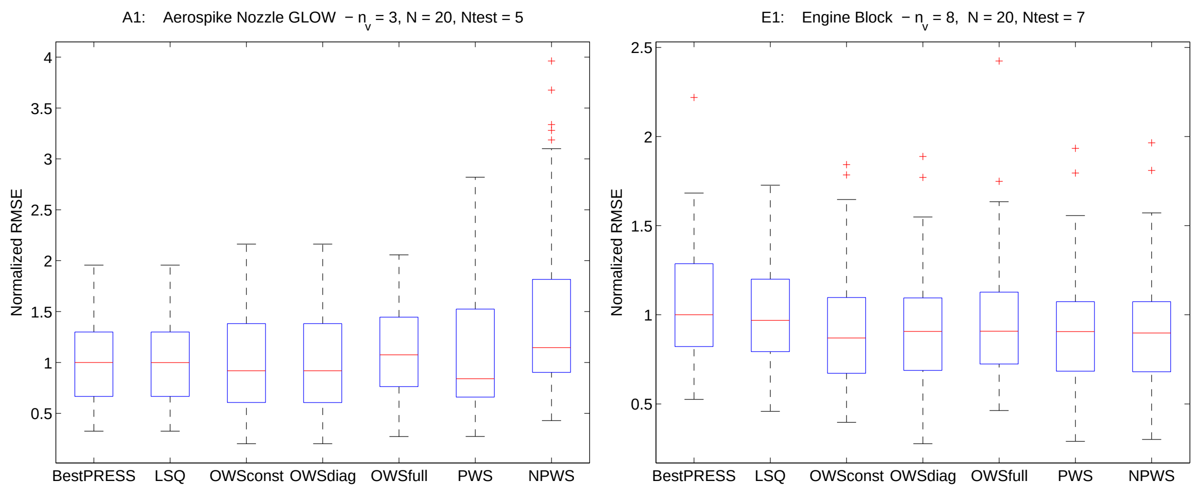
The results of engineering problems are presented in Table 11 and Figure 61. These results confirm the general trends observed with the tests with analytical functions presented in the previous section. That is, the accuracy of LSQ, measured by means of median, mean and standard error of $RMSE$ over 100 runs, are in the same level of magnitude of PRESS based methods.

Table 11: Results of engineering problems.

	BestPRESS			LSQ			OWSconst			OWSdiag			OWSfull			PWS			NPWS		
ID	$\bar{\bar{x}}$	$\bar{x}$	s	$\bar{\bar{x}}$	$\bar{x}$	s	$\bar{\bar{x}}$	$\bar{x}$	s	$\bar{\bar{x}}$	$\bar{x}$	s	$\bar{\bar{x}}$	$\bar{x}$	s	$\bar{\bar{x}}$	$\bar{x}$	s	$\bar{\bar{x}}$	$\bar{x}$	s
A1	1.00	0.98	0.36	1.00	0.98	0.37	0.92	0.97	0.43	0.92	0.97	0.43	1.07	1.10	0.44	0.84	1.12	0.66	1.15	1.50	0.90
A2	1.00	1.04	0.46	1.01	1.05	0.46	1.04	1.05	0.44	1.04	1.05	0.44	1.23	1.39	0.88	1.79	1.92	0.73	4.08	4.91	3.02
A3	1.00	1.01	0.41	1.28	3.34	3.54	0.98	1.04	0.48	0.98	1.05	0.48	0.95	1.06	0.45	1.05	1.25	0.60	1.41	1.65	0.78
C1	1.00	1.01	0.30	1.05	1.49	0.99	0.99	1.02	0.27	1.03	1.03	0.27	0.92	0.96	0.29	1.18	1.18	0.31	1.42	1.43	0.37
C2	1.00	1.02	0.26	1.06	1.44	0.88	0.97	1.01	0.25	1.02	1.03	0.25	0.97	0.99	0.23	1.18	1.20	0.36	1.49	1.58	0.44
C3	1.00	1.05	0.22	1.08	1.16	0.44	1.00	1.04	0.25	0.99	1.04	0.30	1.08	1.11	0.26	1.05	1.09	0.34	1.16	1.14	0.38
E1	1.00	1.05	0.32	0.97	1.00	0.29	0.87	0.90	0.31	0.91	0.92	0.30	0.91	0.96	0.31	0.91	0.91	0.30	0.90	0.90	0.30
E2	1.00	0.96	0.30	1.07	1.45	1.21	0.98	0.93	0.29	1.09	1.08	0.36	0.51	0.53	0.14	1.45	1.51	0.51	1.55	1.61	0.52

Obs.1: $\bar{\bar{x}}$, $\bar{x}$ and s are respectively: median, mean and standard deviation of $RMSE$ over 100 runs.
Obs.2: The values of $\bar{\bar{x}}$, $\bar{x}$ and s are normalized in relation of the median $RMSE$ measured for BestPRESS in each case.

Figure 61 presents the boxplots for the normalized $RMSE$ for the case problems A1 and E1 (see Table 11). As can be seen, the dispersion of $RMSE$ over 100 runs is comparable among the all the WAS methods studied. The other cases were not shown only for brevity, but the overall dispersion behavior of $RMSE$ is similar to the achieved for A1 and E1 cases.



(a) (b)

Figure 61: Boxplots of normalized $RMSE$ for engineering problems A1 (a) and E1 (b), respective to Table 11.

The Table 12 presents the absolute CPU time spent for constructing the WAS models for each one of the engineering problems studied. As can be observed, the time spent in LSQ remains almost constant, since in all the engineering problems the number of sampling points considered are the same ($N = 20$). On the other hand, the cost of PRESS based methods are affected also by the number of variables in the model. In the worst case, A1, LSQ method performed 34 times faster. In the best case, E2, LSQ performed 46 times faster than the other methods.

Table 12: CPU times (seconds) for engineering problems.

ID	BestPRESS	LSQ	OWSconst	OWSdiag	OWSfull	PWS	NPWS
A1	5.7348	0.1686	5.7480	5.7410	5.7346	5.7388	5.7329
C1	6.1669	0.1643	6.1892	6.1737	6.1664	6.1719	6.1650
E1	5.8532	0.1635	5.8725	5.8625	5.8532	5.8561	5.8525
E2	7.5249	0.1650	7.5431	7.5348	7.5263	7.5275	7.5236

One should argue that an order of 5 or 7 seconds spent in WAS calculation for PRESS based methods is not significant. On the other hand, this impact should be expressive in optimization contexts, specially when new sampling points (infill points) are iteratively included in the model to reduce the approximation error and improve the optimization results. It is not rare an iterative optimization problem to require an order of 100 or more infill iterations to achieve an acceptable optimum result (see for instance Forrester et al (2008)). If we need, for example, to construct the metamodel problem E2 in 100 optimization iterations, the cost would be 16.50s considering the LSQ method, against 752.49s for BestPRESS. Therefore, this difference of CPU time should not be neglected.

In summary, the results corroborate that, in practical applications, LSQ method should be considered as an alternative of the PRESS based methods, since it is comparable in terms of accuracy and it performs much faster than the other WAS methods available.

F Multicollinearity and Least Squares

F.1 The Sources of Multicollinearity

The issue of multicollinearity in least squares regression is well known in statistics and related areas and the research in this front remounts at least to the decade of 1950. See for example Björk (1996) and Montgomery et al (2006) and the list of references therein for a broader perspective on this subject.

By definition, the least squares problem is based on the assumption that the k regressors x_i, or *predictor variables*, in the simple linear case of Eq. (18), are *mutually orthogonal.* In other words, it is assumed in advance that there is no linear relationship among the predictor variables.

In matrix form, the least squares problem can be stated as

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon} \ , \tag{43}$$

where $\mathbf{y}$ is a $(N \times 1)$ vector of responses; $\mathbf{X}$ is a $(N \times p)$ matrix of the regressor variables; $\boldsymbol{\beta}$ is a $(p \times 1)$ vector of unknown coefficients; and $\boldsymbol{\varepsilon}$ is a $(p \times 1)$ vector of random errors, that are assumed to be normally and independently distributed, with zero mean and finite variance, i.e., $\varepsilon_i \sim$ NID$(0, \sigma^2)$. In this form, N represents the number of observations (or samples) and $p = k$ when the intercept term β_0 is considered zero and $p = (k + 1)$, otherwise.

One possible solution for Eq. (43) is the standard least squares estimator, i.e.,

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} \ , \tag{44}$$

that has the following properties:

(a) *Unbiasedness*:

$$\text{Bias}\left(\hat{\boldsymbol{\beta}}\right) \equiv E\left[\hat{\boldsymbol{\beta}}\right] - \boldsymbol{\beta} = 0 \quad \Rightarrow \quad E\left[\hat{\boldsymbol{\beta}}\right] = \boldsymbol{\beta} \ ; \tag{45}$$

(b) *Variance*:

$$\begin{aligned} \text{Var}\left(\hat{\boldsymbol{\beta}}\right) &\equiv E\left[\hat{\boldsymbol{\beta}}\hat{\boldsymbol{\beta}}^T\right] - E\left[\hat{\boldsymbol{\beta}}\right]\left(E\left[\hat{\boldsymbol{\beta}}\right]\right)^T \\ &= \sigma^2(\mathbf{X}^T\mathbf{X})^{-1} \ , \qquad \text{where} \qquad (46) \\ \sigma^2 &\approx \hat{\sigma}^2 = \frac{\mathbf{y}^T\mathbf{y} - \hat{\boldsymbol{\beta}}^T\mathbf{X}^T\mathbf{y}}{N-p} \quad ; \end{aligned}$$

(c) *Mean Squared Error*:

$$\begin{aligned} \text{MSE}\left(\hat{\boldsymbol{\beta}}\right) &\equiv E\left[\left\|\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}\right\|^2\right] \\ &= \text{tr}\left\{\text{Var}\left(\hat{\boldsymbol{\beta}}\right)\right\} + \left\|\text{Bias}\left(\hat{\boldsymbol{\beta}}\right)\right\|^2; \end{aligned} \tag{47}$$

(d) *Gauss-Markov Theorem*: The least squares estimator $\hat{\boldsymbol{\beta}}$ is the best linear unbiased estimator (BLUE) of $\boldsymbol{\beta}$.

For proofs and details on these properties see Montgomery et al (2006) or Björk (1996).

Unfortunately, in most applications the assumption of mutually orthogonal does not hold and the final regression model can be misleading or erroneous. Thus, when there are linear or near-linear dependencies among the regressors, the problem of *multicollinearity* arises. This is due to the fact that the so called *correlation matrix* $\mathbf{X}^T\mathbf{X}$ has rank lower than p and for consequence its inverse does not exist anymore. In this case, the least squares estimate $\hat{\boldsymbol{\beta}}$ becomes numerically unstable.

This instability of the coefficients can be explained by the variance definition. If the matrix $\mathbf{X}^T\mathbf{X}$ has linear dependence among columns (i.e., multicollinearity), then the variance of the coefficients can increase rapidly or become infinite and by consequence the prediction will be poor.

Among the several sources of multicollinearity, the primary ones are: (i) the data collection method (size and distribution of sampling points) and (ii) model overdefined or with redundant variables. During the last decades, several methods have been devised for dealing with multicollinearity in least squares problems (see Montgomery et al (2006), Chap. 11). In general the techniques include gathering additional data and some kind of modification in the the way that the coefficients $\hat{\boldsymbol{\beta}}$ are estimated, in order to reduce the prediction errors induced by multicollinearity.

F.2 Prior Model Selection

In our context, the main source of multicollinearity is due to the fact the models $\hat{y}_i(\mathbf{x})$ tend to be very similar since all of them are trying to match the true response $y(\mathbf{x})$, as best as possible, therefore the problem is overdefined on its nature. In addition, this situation can be worsened if the sampling points are not well distributed in the design space.

In this way, by assuming that we have a fairly good design space distribution, the first option is to conduct a prior selection and remove the most redundant models in the set $[\hat{y}_1(\mathbf{x}), \hat{y}_2(\mathbf{x}), \dots, \hat{y}_M(\mathbf{x})]$, by means of some heuristic method.

One quick way, for instance, can be by using the concept of correlation, i.e., by defining the *pairwise correlation matrix* $\mathbf{R} = [r^2(\hat{y}_i, \hat{y}_j)]$, where $r(X_i, Y_i)$ is the *sample linear correlation coefficient*, that is,

$$r(X_i, Y_i) = \frac{\sum_{i=1}^{N} (X_i - \bar{X})(Y_i - \bar{Y})}{\left[\sum_{i=1}^{N} (X_i - \bar{X})^2 \sum_{i=1}^{N} (Y_i - \bar{Y})^2\right]^{\frac{1}{2}}} \tag{48}$$

with

$$\bar{X} = \frac{1}{N} \sum_{i=1}^{N} X_i,$$

for any two random vectors X_i and Y_i, of size N.

In this way, $R_{ii} = 1$ and $(0 \le R_{ij} \le 1)$, for $i \ne j$. Therefore, we can easily identify the most correlated models based on a threshold, say for example $(R_{ij} \ge 0.8)$, and verify if it is possible to eliminate the less significant model in the pair ij from the set, before creating the ensemble.

This heuristic approach can be useful when we have a large set of models, thus we can rapidly identify the most correlated pairs and remove the poorest ones in terms of accuracy in advance. It is worth noting that, specially for small sets, this criterion must be carefully used since interpolating or highly accurate models can lead to $(R_{ij} \to 1.0)$ as well, and of course cannot be discarded blindly.

Other diagnostics for multicollinearity exist in the least squares literature. Most of them are based on the examination of the correlation matrix $\mathbf{X}^T\mathbf{X}$, namely: correlation coefficients, determinant, eigensystem analysis, VIF (variance inflation factors), etc. We will not present them here, since at the end of the day all these diagnostic measures are useful to estimate pairwise correlation and not more than that. For example, it is possible to identify pairs of highly correlated models, but if the collinearity is among more than two models it cannot be identified by a simple inspection. In addition, as we already discussed, pure collinearity does not mean directly that models are not significant in terms of accuracy or predictability for the ensemble. Refer to Chap. 11 of Montgomery et al (2006) for a detailed discussion on this subject.

F.3 Gathering Additional Data

As reported in Montgomery et al (2006), one of the best methods to reduce the sources of multicollinearity in least squares is collecting additional data. The idea is first to understand the distribution of points in the design space and add more sampling points in the non-populated areas, in order to avoid concentrations along lines and therefore break-up multicollinearity.

In many cases, unfortunately, collecting additional data is costly or even impossible. In other cases, collecting additional data is not a viable solution to the multicollinearity problem because its source is due to constraints on the model or in the population. This is the case of ensemble of metamodels. All the models $\hat{y}_i$ are trying to approximate the true response y as best as possible, or exactly in the case of interpolation. Therefore, multicollinearity will come up naturally since the columns formed by the metamodels $\hat{y}_i$ tend to be quite similar.

F.4 Variable Selection Methods in Regression

A central problem in least squares regression is related to the definition of the *best set* of variables or predictors to build the model. It is desired to define a *parsimonious model*, i.e., the simpler regression model that represent the problem at hand, as always as possible. A parsimonious model is easier to interpret, to collect data and, in addition, it is less prone to redundancies that induce linear dependencies and multicollinearity and reduce accuracy and predictability.

It is well known that the key driving question in all the variable selection methods is: *"How to include or exclude variables in a least squares model in order to achieve the desired accuracy and be parsimonious at same time?"*

During the decades several methods have been devised, implemented and tested in an attempt to answer this question. In this sense, let us briefly explain the concept of balancing bias and variance in least squares approximation.

F.4.1 The Bias and Variance Dilemma

As remarked by Montgomery et al (2006), the Gauss-Markov property assures that the estimator $\hat{\boldsymbol{\beta}}$ has minimum error, in the least squares sense, among all unbiased linear estimators, but there is no guarantee that its variance will be small.

As we discussed previously, when the method of least squares is applied to nonorthogonal data, very inaccurate estimates of $\boldsymbol{\beta}$ can be obtained, due to the inflation of the variance. This implies that the absolute values of the coefficients are very unstable and may dramatically change in sign and magnitude by small variations in the design matrix $\mathbf{X}$.

One way to mitigate this issue is to relax the requirement that the estimator of $\boldsymbol{\beta}$ be unbiased. Let us assume that we can find a $\hat{\boldsymbol{\beta}}^*$ in such a way that

$$\hat{\boldsymbol{\beta}}^* = \hat{\boldsymbol{\beta}} - \boldsymbol{\delta}, \qquad \text{with} \qquad \|\boldsymbol{\delta}\| < \left\|\hat{\boldsymbol{\beta}}\right\|, \qquad \text{and} \qquad \mathrm{E}\left[\boldsymbol{\delta}\right] = \delta, \tag{49}$$

then the bias will be

$$\mathrm{Bias}\left(\hat{\boldsymbol{\beta}}^*\right) = \mathrm{E}\left[\hat{\boldsymbol{\beta}} - \boldsymbol{\delta}\right] - \boldsymbol{\beta} \qquad \Rightarrow \qquad \mathrm{Bias}\left(\hat{\boldsymbol{\beta}}^*\right) = -\delta \tag{50}$$

and, by assuming that $\hat{\boldsymbol{\beta}}$ and $\boldsymbol{\delta}$ are independent, then

$$\mathrm{Var}\left(\hat{\boldsymbol{\beta}}^*\right) = \mathrm{Var}\left(\hat{\boldsymbol{\beta}} - \boldsymbol{\delta}\right) = \mathrm{Var}\left(\hat{\boldsymbol{\beta}}\right) - \mathrm{Var}\left(\boldsymbol{\delta}\right). \tag{51}$$

In addition, the MSE will become

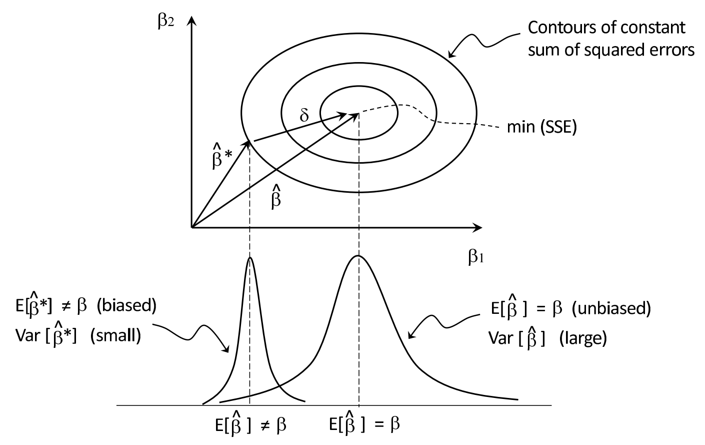
$$\begin{aligned} \mathrm{MSE}\left(\hat{\boldsymbol{\beta}}^*\right) &= \mathrm{tr}\left\{\mathrm{Var}\left(\hat{\boldsymbol{\beta}}\right)\right\} - \mathrm{tr}\left\{\mathrm{Var}\left(\boldsymbol{\delta}\right)\right\} + \|\boldsymbol{\delta}\|^2 \\ &= \mathrm{E}\left[\left\|\hat{\boldsymbol{\beta}} - \boldsymbol{\beta} - \boldsymbol{\delta}\right\|^2\right] \\ &= \mathrm{E}\left[2\left\|\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}\right\|^2 + 2\left\|\boldsymbol{\delta}\right\|^2 - \left\|\hat{\boldsymbol{\beta}} - \boldsymbol{\beta} + \boldsymbol{\delta}\right\|^2\right] \\ &= 2\mathrm{MSE}\left(\hat{\boldsymbol{\beta}}\right) + f\left(\|\boldsymbol{\delta}\|^2\right) \end{aligned} \tag{52}$$

by using the parallelogram law for vector norms and the linearity of the expectation operator.

In summary, it can be concluded that, by allowing a small amount $\boldsymbol{\delta}$ of bias in $\hat{\boldsymbol{\beta}}^*$, the variance of $\hat{\boldsymbol{\beta}}^*$ will be smaller than $\hat{\boldsymbol{\beta}}$. On the other hand, the mean squared error at the data may increase rapidly as a function of the level of bias induced. If the effect of increasing bias is smaller than the effect of reducing variance then it is possible to reduce the error. Therefore, by controlling the size of $\hat{\boldsymbol{\beta}}$, it is possible to control the stability and error level of the solution by balancing bias and variance.

In Figure 62, a geometrical interpretation on this behavior of the solution of $\hat{\boldsymbol{\beta}}$ in terms of bias and variance is presented for a generic problem with two variables, i.e., $\boldsymbol{\beta} = [\beta_1, \beta_2]^T$. The idea behind this behavior is by choosing a smaller estimator $\hat{\boldsymbol{\beta}}^*$, then the variance will be smaller. As consequence, the price for reducing variance will be always by adding bias in the solution, i.e., the mean squared error will not be the minimum anymore at the sampling points.

In practical terms, it means that if one chooses a biased estimator $\hat{\boldsymbol{\beta}}^*$, by increasing the



β_2
Contours of constant sum of squared errors
δ
min (SSE)
$\hat{\beta}^*$
$\hat{\beta}$
β_1
$E[\hat{\beta}^*] \neq \beta$ (biased)
Var $[\hat{\beta}^*]$ (small)
$E[\hat{\beta}] = \beta$ (unbiased)
Var $[\hat{\beta}]$ (large)
$E[\hat{\beta}] \neq \beta$
$E[\hat{\beta}] = \beta$

Figure 62: An illustration of bias and variance dilemma in least squares. The BLUE estimator has minimal MS error but the associated variance is large. On the other hand, by accepting some bias or in other words *shrinking* the vector $\hat{\boldsymbol{\beta}}$, the MSE increases and the variance is reduced (improved stability). Adapted based on Montgomery et al (2006).

mean squared error at sampling points, the variance will be reduced and the sensitivity of the regression coefficients to changes on the data will be also reduced (i.e., less sensitivity to noise or perturbations in the components of matrix $\mathbf{X}$). Therefore, with more stable coefficients, the overfitting can be reduced and the accuracy of predictions for future data will increase as well.

The possible ways to reduce the magnitude of the vector $\hat{\boldsymbol{\beta}}$ are mainly two: (i) by removing/combining variables from the scope of the model, or by forcing some of the $\hat{\beta}_i = 0$; and (ii) by reducing (shrinking) the size of the vector $\hat{\boldsymbol{\beta}}$.

Based on these two central ideas, most of the methods summarized in Section F.5 were devised in order to find a solution on how to trade-off between bias and variance, and improve accuracy and predictability for a given set of variables in a problem least squares problem.

Finally, variable selection methods is a large front of research in least squares approximation field. Miller (2002) presented an extensive review on variable selection in regression problems and this is still a subject of active research, as can bee seen in the recent publications, for instance Ng (2012) which states that: “The variable selection is by no means solved.” and Scheipl et al (2013) that reinforces that there is still a wide and open field for future research in variable and function selection in multivariate regression.

F.5 Least Squares Variants

F.5.1 Stepwise Regression

Several heuristic procedures and algorithms have been developed to add and/or delete variables in a regression model to control the accuracy and predictability. In Miller (2002) it is presented an extensive discussion on these methods.

The idea is to start from a large set of possible variables and to find a small subset of variables (or candidate subsets) that reduces the modeling errors. Then this field of research is known as: *subset selection in regression.*

By starting from the *full model* on the Equation (43), let us assume that there are p candidate regressors x_1, x_2, $\cdots$, x_p. In addition, let r be the number of regressors that are to be deleted from the full model. Therefore, the regression equation can be written as

$$\mathbf{y} = \mathbf{X}_s\boldsymbol{\beta}_s + \mathbf{X}_r\boldsymbol{\beta}_r + \boldsymbol{\varepsilon} \ , \tag{53}$$

where $\mathbf{X}$ has been partitioned into $\mathbf{X}_s$, a $(N \times s)$ matrix whose columns represent the regressors to be retained; and $\mathbf{X}_r$, a $(N \times r)$ matrix whose columns represent the regressors to be deleted from the full model. In the same sense, $\boldsymbol{\beta}$ is partitioned accordingly in $\boldsymbol{\beta}_s$ and $\boldsymbol{\beta}_r$.

Now, for the *subset* model,

$$\mathbf{y} = \mathbf{X}_s\boldsymbol{\beta}_s + \boldsymbol{\varepsilon} \ , \tag{54}$$

the least squares estimate is, similarly to the full model derivation,

$$\hat{\boldsymbol{\beta}}_s = \left(\mathbf{X}_s^T\mathbf{X}_s\right)^{-1}\mathbf{X}_s^T\mathbf{y} \ . \tag{55}$$

By analyzing the properties of the estimator $\hat{\boldsymbol{\beta}}_s$, it can be demonstrated that

$$\begin{aligned} \mathrm{E}\left[\hat{\boldsymbol{\beta}}_s\right] &= \hat{\boldsymbol{\beta}}_s + \left(\mathbf{X}_s^T\mathbf{X}_s\right)^{-1}\mathbf{X}_s^T\mathbf{X}_r\hat{\boldsymbol{\beta}}_r \\ &= \hat{\boldsymbol{\beta}}_s + \mathbf{A}\hat{\boldsymbol{\beta}}_r \end{aligned} \tag{56}$$

where $\mathbf{A} = \left(\mathbf{X}_s^T\mathbf{X}_s\right)^{-1}\mathbf{X}_s^T\mathbf{X}_r$ is called *aliasing* matrix. Thus $\hat{\boldsymbol{\beta}}_s$ is a biased estimate of $\boldsymbol{\beta}_s$, i.e., $\mathrm{Bias}\left(\boldsymbol{\beta}_s\right) = \mathbf{A}\hat{\boldsymbol{\beta}}_r$, unless the deleted variables $\boldsymbol{\beta}_r$ are zero or the retained variables are orthogonal to the deleted variables (i.e., $\mathbf{X}_s^T\mathbf{X}_r = \mathbf{0}$).

In case of variance, it follows that

$$\mathrm{Var}\left(\hat{\boldsymbol{\beta}}_s\right) = \sigma^2\left(\mathbf{X}_s^T\mathbf{X}_s\right)^{-1} . \tag{57}$$

By the fact that the variance is a positive semidefinite matrix and provided that

$$\mathbf{v}^T\left(\mathbf{A}+\mathbf{B}\right)\mathbf{v} \quad \geq \quad \mathbf{v}^T\mathbf{A}\mathbf{x} \qquad \forall \mathbf{v} \neq \mathbf{0} \ , \tag{58}$$

and $\mathbf{X}$ can be rewritten as

$$\begin{aligned}\mathbf{X} &= [\mathbf{X}_s \;\; \mathbf{0}] + [\mathbf{0} \;\; \mathbf{X}_r] \\ &= \mathbf{S} + \mathbf{R}\end{aligned} \tag{59}$$

then, it can be concluded that,

$$\mathbf{v}^T \left(\mathbf{X}^T\mathbf{X}\right) \mathbf{v} \;\; \geq \;\; \mathbf{v}^T \left(\mathbf{S}^T\mathbf{S}\right) \mathbf{v} \tag{60}$$

which leads to

$$\text{Var}\left(\hat{\boldsymbol{\beta}}\right) \;\; \geq \;\; \text{Var}\left(\hat{\boldsymbol{\beta}}_s\right) \tag{61}$$

That is, the variance of the least squares estimates of the parameters in the full model Var $\left(\hat{\boldsymbol{\beta}}\right)$ are greater than or equal to to the variances of the corresponding parameters in the subset model, Var $\left(\hat{\boldsymbol{\beta}}_s\right)$. Consequently, deleting variables from the full model never increases the variances of the estimates of subset models.

Finally, the mean squared error for the subset model is

$$\text{MSE}\left(\hat{\boldsymbol{\beta}}_s\right) = \text{tr}\left\{\sigma^2\left(\mathbf{X}_s^T\mathbf{X}_s\right)^{-1}\right\} \quad + \quad \left\|\mathbf{A}\hat{\boldsymbol{\beta}}_r\right\|^2 \tag{62}$$

and again, since the bias is introduced by deleting variables, the mean squared error for the subset model will be lower than the full model, if and only if the effect of reduction in variance is higher than the effect of increasing bias. In this way, the search for smaller subsets with lower variance and possible lower MSE than the full model makes sense, although there is no proof in advance that there exist at least one subset model that satisfies this expectation.

Because evaluating all possible subsets can be computationally burdensome, various methods have been developed for evaluating only a small number of subset regression models by adding or deleting variables one at a time, step by step. These methods are generally referred to as *stepwise-type procedures.* There are in general three categories: (1) *forward selection*, starting from no variables and adding one at time; (2) *backward elimination*, starting from the full model and deleting variables one at a time; and (3) *stepwise regression*, a common combination of procedures (1) and (2).

Based on specific statistical assumptions, several *stopping rules* were developed for this class of algorithms. Efroymson (1960) was the first to use the term "stepwise regression" and his method is based on the F-statistic distribution to define the significance level to add or delete variables, and it is still used today in several computer codes for least squares regression.

The *Efroymson's Algorithm* can be summarized as follows, according to Miller (2002):

Let us first define the *residual sum of squares* as

$$\begin{aligned} RSS &= \sum_{i=1}^{N} e_i^2 \\ &= \sum_{i=1}^{N} (y_i - \hat{y}_i)^2 \ , \end{aligned} \tag{63}$$

where y_i is the true response and $\hat{y}_i$ is the fitted value for the i-th data site.

(a) *Addition*: Let RSS_s denote the residual sum of squares with s variables in the model. Suppose the smallest RSS which can be obtained by adding another variable to the present set, is RSS_{s+1}. The ratio

$$R_e = \frac{RSS_s - RSS_{s+1}}{RSS_{s+1}/\left(N - s - 2\right)} \tag{64}$$

is calculated and compared with an 'F-to-enter' value, F_e. If R_e is greater than F_e, the variable is added to the selected set.

(b) *Deletion*: With s variables, let RSS_{s-1} be the smallest RSS that can be obtained after deleting any variable from the previously selected ones. The ratio

$$R_d = \frac{RSS_{s-1} - RSS_s}{RSS_s/\left(N - s - 1\right)} \tag{65}$$

is calculated and compared with an 'F-to-delete' value, F_d. If R_d is less than F_d, the variable is deleted from the selected set.

(d) *Level of Significance*: The criteria F_e and F_d are calculated based on the inverse of F cumulative distribution function, i.e., $F^{-1}_{(\alpha,\ \nu_1,\ \nu_2)}$, with numerator degrees of freedom ν_1 and denominator degrees of freedom ν_2, for the corresponding level of significance $\alpha \times 100\%$. If, for instance, for a variable to be added, it is considered 95% percentile of F distribution (or $\alpha = 5\%$), then $F_e = F^{-1}_{(0.05,\ s+1,\ N-s-2)}$. So, if R_e is greater than F_e, it is the same to say that p-value $> 5\%$, and therefore *we fail to reject* H_0 in the following hypotheis test:

$$\begin{cases} H_0\text{:} & \text{the variable } x_{s+1} \text{ is insignificant, i.e., } \beta_{s+1} = 0. \\ H_1\text{:} & \text{the variable } x_{s+1} \text{ is not insignificant, i.e., } \beta_{s+1} \neq 0. \end{cases} \tag{66}$$

Henceforth, we conclude with more than 95% confidence level that the variable *is significant* to be added in the selected subset model.

The algorithm starts with no variables in the model and then moves *forward* by adding all the variables considered significant to the selected subset. In the sequence the algorithm moves *backward* by testing if any of the previously selected variables can be deleted without appreciably increasing the residual sum of squares. It is common to use the level of significance ($0.01 < \alpha < 0.05$), and if one needs to find a more parsimonious model (smaller subset), then set ($\alpha_e < \alpha_d$) and the addition will be more strict than deletion of variables.

Other popular stepwise regression methods have been proposed, motivated by different stopping criteria, for instance, *apud* Fang et al (2005): Mallows C_p criterion, (Mallows, 1973); AIC - Akaike Information Criterion, Akaike (1974); BIC - Bayesian Information Criterion, Schwarz (1978); φ-Criterion, Hannan and Quinn (1979) and Shibata (1984); and RIC - Residual Information Criterion, Foster and George (1994).

For more details on these algorithms and variations see Miller (2002), Montgomery et al (2006) and in special Fang et al (2005), that presents a good set of engineering computational examples and comparisons based on these methods for subset selection in regression.

F.5.2 Ridge Regression

One of the most famous class of biasing estimators is the *ridge regression*, originally proposed in the two companion papers Hoerl and Kennard (1970a) and Hoerl and Kennard (1970b). The ridge estimator is found by solving a modified version of the normal equations. In this approach, the ridge estimator $\hat{\boldsymbol{\beta}}_R$ is the solution of

$$\hat{\boldsymbol{\beta}}_R = \left(\mathbf{X}^T\mathbf{X} + k\mathbf{I}_p\right)^{-1} \mathbf{X}^T\mathbf{y} \ , \tag{67}$$

where $k \geq 0$ is a constant parameter to control the level of bias-variance in the solution and $\mathbf{I}_p$ is the $(p \times p)$ identity matrix. As can be noted, when $k = 0$ the solution is the unbiased least squares estimator.

It can be noted as well that the ridge estimator is a linear transformation of the standard least squares estimator, i.e., $\hat{\boldsymbol{\beta}}_R = \mathbf{Z}_k\hat{\boldsymbol{\beta}}$, where $\mathbf{Z}_k$, given by

$$\mathbf{Z}_k = \left(\mathbf{X}^T\mathbf{X} + k\mathbf{I}_p\right)^{-1} \left(\mathbf{X}^T\mathbf{X}\right) \ , \tag{68}$$

is the matrix in the linear transformation (function of k), since $\mathbf{y} = \mathbf{X}\hat{\boldsymbol{\beta}}$.

Thus, k controls the size of $\hat{\boldsymbol{\beta}}_R$, by scaling (or shrinking) the vector $\hat{\boldsymbol{\beta}}$, in order to induce bias and reduce the variance. In this sense the class of biased estimators are also called *shrinkage* estimators.

By rewriting the equation for mean squared error, it can be shown that

$$\begin{aligned}
\mathrm{MSE}\left(\hat{\boldsymbol{\beta}}_R\right) &= \mathrm{tr}\left\{\mathrm{Var}\left(\hat{\boldsymbol{\beta}}\right)\right\} + \left\|\mathrm{Bias}\left(\hat{\boldsymbol{\beta}}\right)\right\|^2 \\
&= \mathrm{tr}\left\{\sigma^2 \mathbf{Z}_k \left(\mathbf{X}^T\mathbf{X} + k\mathbf{I}_p\right)^{-1}\right\} \quad + \quad k^2 \boldsymbol{\beta}^T \left(\mathbf{X}^T\mathbf{X} + k\mathbf{I}_p\right)^{-2} \boldsymbol{\beta} \qquad (69) \\
&= \sigma^2 \sum_{j=1}^{p} \frac{\lambda_j}{\left(\lambda_j + k\right)^2} \quad + \quad k^2 \boldsymbol{\beta}^T \left(\mathbf{X}^T\mathbf{X} + k\mathbf{I}_p\right)^{-2} \boldsymbol{\beta}
\end{aligned}$$

where λ_1, λ_2, $\cdots$, λ_p are the eigenvalues of the matrix $\mathbf{X}^T\mathbf{X}$. Then, if $k > 0$, the variance in $\hat{\boldsymbol{\beta}}_R$ monotonically decreases as the bias increases. In addition, Hoerl and Kennard proved that exists a nonzero k for which the MSE($\hat{\boldsymbol{\beta}}_R$) will be less than the variance of the least squares estimator $\hat{\boldsymbol{\beta}}$. Therefore, the challenge is to find k in a such way that the reduction in variance term is greater than the bias term, then MSE $\left(\hat{\boldsymbol{\beta}}_R\right)$ will be reduced as well.

Hoerl and Kennard originally proposed the method by graphical inspection of the behavior of the coefficients with respect to increasing values of k, i.e., *ridge trace inspection.* Ridge trace inspection is a good method for a small number of coefficients but it is not practical to solve problems with large number of variables in an automatic and systematic way. In addition, the inspection of ridge trace is a subjective procedure requiring judgment form the analyst.

An analytical way to estimate of k was suggested by Hoerl et al (1975), that is

$$k = \frac{p\hat{\sigma}^2}{\hat{\boldsymbol{\beta}}^T\hat{\boldsymbol{\beta}}} \ , \qquad (70)$$

where $\hat{\boldsymbol{\beta}}$ and $\hat{\sigma}^2$ are calculated based on the ordinary least squares solution. In a subsequent paper, Hoerl and Kennard (1976) proposed an iterative algorithm based on the Equation (70).

There are many other variants proposed in the literature for choosing k. Even though, as noted by Montgomery et al (2006), there is no guarantee that all these methods are superior to straightforward inspection of the ridge trace.

In this sense, the generalized cross-validation statistic (GCV) procedure by Golub et al (1979) is one of the most accepted for automation, i.e.,

$$\mathrm{GCV}\left(k\right) = \frac{1}{N} \frac{\left\|\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\beta}}\right\|}{\left[1 - e(\kappa)/N\right]^2} \ , \qquad (71)$$

where

$$e\left(k\right) = \mathrm{tr}\left[\mathbf{X}\left(\mathbf{X}^T\mathbf{X} + k\mathbf{I}\right)^{-1}\mathbf{X}^T\right] \qquad (72)$$

and k is selected to be

$$k = \arg\min_k \mathrm{GCV}\left(k\right) \ . \qquad (73)$$

The minimization procedure is done by computing GCV for a discrete set of points in the interval $(0 < k < 1)$, therefore the final estimator $\hat{\boldsymbol{\beta}}_R$ is the one that has the lowest GCV(k) score.

Once k is defined or selected, it is possible to modify the ridge matrices definition and then use a standard least squares algorithm, as follows,

$$\mathbf{X}_A = \begin{bmatrix} \mathbf{X} \\ \sqrt{k}\mathbf{I}_p \end{bmatrix}, \qquad \mathbf{y}_A = \begin{Bmatrix} \mathbf{y} \\ \mathbf{0}_p \end{Bmatrix}, \tag{74}$$

with $\mathbf{I}_p$ and $\mathbf{0}_p$ respectively the identity matrix of order $p \times p$ and null vector of order $p \times 1$, and then

$$\hat{\boldsymbol{\beta}}_R = \left(\mathbf{X}_A^T\mathbf{X}_A\right)^{-1}\mathbf{X}_A^T\mathbf{y}_A = \left(\mathbf{X}^T\mathbf{X} + k\mathbf{I}_p\right)^{-1}\mathbf{X}^T\mathbf{y} \ . \tag{75}$$

It is worth noting that the ridge solution can be viewed as a *constrained* least squares in the form

$$\begin{array}{ll} \min\limits_{\boldsymbol{\beta}} \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2 \\ \text{subject to} \quad \boldsymbol{\beta}^T\boldsymbol{\beta} \leq d^2 \end{array}, \tag{76}$$

where the radius d depends on k.

Finally, as remarked by Montgomery et al (2006), the ridge estimate will not necessarily provide the best “fit” to the data, but this is not the real concern, since the main interest is on reducing multicollinearity and to obtain a stable set of parameter estimates. The ridge estimates may result in an equation that does a better job of prediction for future observations than would the ordinary least squares, although there is no guarantee nor conclusive proof that this will happen in fact.

F.5.3 Principal Component Regression

Another approach that can be used to generate biased least squares estimators is known as *principal component regression.* A deep and extensive discussion on principal components analysis can be found in Jolliffe (2002).

Let us define a linear matrix transformation such that

$$\mathbf{Z} = \mathbf{X}\mathbf{V}, \tag{77}$$

where $\mathbf{V} = [\mathbf{V}_1 \ \ \mathbf{V}_2 \ \ \cdots \ \ \mathbf{V}_p]$ is a $(p \times p)$ matrix whose k-th column $\mathbf{V}_k$ is the k-th eigenvector of $\mathbf{X}^T\mathbf{X}$. In this way, $\mathbf{Z} = [\mathbf{Z}_1 \ \ \mathbf{Z}_2 \ \ \cdots \ \ \mathbf{Z}_p]$ is a $(p \times p)$ matrix whose k-th column $\mathbf{Z}_k$ is the k-th principal component (PC) of $\mathbf{X}$. In addition, it can be shown that

$$\mathbf{Z}^T\mathbf{Z} = \mathbf{V}^T\mathbf{X}^T\mathbf{X}\mathbf{V} = \boldsymbol{\Lambda} \ , \tag{78}$$

where $\mathbf{\Lambda}$ is a diagonal $(p \times p)$ matrix of eigenvalues of $\mathbf{X}^T\mathbf{X}$, i.e., $\mathbf{\Lambda} = \operatorname{diag}(\lambda_1, \ \lambda_2, \ \cdots \ , \ \lambda_p)$.

Since $\mathbf{V}$ is orthogonal, then $\mathbf{X}\boldsymbol{\beta}$ can be rewritten as $\mathbf{X}\mathbf{V}\mathbf{V}^T\boldsymbol{\beta}$ and the standard regression equation can be rewritten as

$$\mathbf{y} = \mathbf{Z}\boldsymbol{\gamma} + \boldsymbol{\varepsilon} \ , \tag{79}$$

where $\boldsymbol{\gamma}$ is defined as the linear transformation

$$\boldsymbol{\gamma} = \mathbf{V}^T\boldsymbol{\beta} \quad \Rightarrow \quad \boldsymbol{\beta} = \mathbf{V}\boldsymbol{\gamma}, \tag{80}$$

and the predictor variables are replaced by their PC in the regression model, with the same transformation, i.e., $\mathbf{z} = \mathbf{V}^T\mathbf{x}$.

In other words, the original vector of variables $\mathbf{x} = (x_1, \ x_2, \ \cdots \ , \ x_p)^T$ are transformed to a new set of variables $\mathbf{z} = (z_1, \ z_2, \ \cdots \ , \ z_p)^T$ as follows

$$\begin{cases} z_1 = \mathbf{V}_1^T\mathbf{x} & \Rightarrow & z_1 = V_{11}x_1 + V_{21}x_2 + \cdots + V_{p1}x_p \\ z_2 = \mathbf{V}_2^T\mathbf{x} & \Rightarrow & z_2 = V_{12}x_1 + V_{22}x_2 + \cdots + V_{p2}x_p \\ \quad\vdots & \vdots & \quad\vdots \\ z_p = \mathbf{V}_p^T\mathbf{x} & \Rightarrow & z_p = V_{1p}x_1 + V_{2p}x_2 + \cdots + V_{pp}x_p \end{cases} , \tag{81}$$

and, in this way, the variable z_i are called principal components (PC) of the original variables x_i.

By analogy, the least squares estimator of $\boldsymbol{\gamma}$ is

$$\hat{\boldsymbol{\gamma}} = \left(\mathbf{Z}^T\mathbf{Z}\right)^{-1}\mathbf{Z}^T\mathbf{y} = \mathbf{\Lambda}^{-1}\mathbf{Z}^T\mathbf{y} \tag{82}$$

and the variance matrix of $\hat{\boldsymbol{\gamma}}$ is

$$\operatorname{Var}(\hat{\boldsymbol{\gamma}}) = \sigma^2\left(\mathbf{Z}^T\mathbf{Z}\right)^{-1} = \sigma^2\mathbf{\Lambda}^{-1}. \tag{83}$$

Thus, a small k-th eigenvalue of $\left(\mathbf{X}^T\mathbf{X}\right)$ means that the variance of the corresponding orthogonal regression coefficient will be large. If all the eigenvalues $\lambda_i = 1$, then the original variables are orthogonal. On the other hand, one λ_i is equal to zero, implies a perfect linear relationship among the variables. Therefore, one or more eigenvalue λ_i near zero indicate that multicollinearity is present and it can destroy the accuracy of the least squares coefficients estimate, since $\operatorname{Var}(\hat{\gamma}_k) = \sigma^2\lambda_k^{-1} \to \infty$.

Therefore, the use of principal component approach can provide many advantages in least squares approximation. By using principal components it is possible to combat multicollinearity by using less than the full set of transformed variables z_i in the regression model.

In their definition, see Jolliffe (2002), the PC are generated in sequence in a way that the first PC, i.e., $z_1 = \mathbf{V}_1^T\mathbf{x}$ is a linear function of elements of $\mathbf{x}$ having maximum variance.

The second PC, $z_2 = \mathbf{V}_2^T\mathbf{x}$, is another linear transformation, uncorrelated with z_1, and having maximum variance, and so on, that the k-th step is a linear function $z_k = \mathbf{V}_k^T\mathbf{x}$ is generated that has maximum variance and it is uncorrelated with all the previously generated PC, i.e., $z_1, z_2, \cdots, z_{k-1}$. Therefore, up to p PC can be generated and it is hoped, in general, that the most variation in $\mathbf{x}$ will be accounted for by s PC, where $(s < p)$. Then it is possible to reduce the complexity of the problem by transforming the original variables to the smallest set of PC able to explain the data.

Figure 63 presents a geometric interpretation of the idea of principal components. In this hypothetical example, the original variables x_1 and x_2 are clearly correlated, since the data dispersion is concentrated around a straight line, which is an evidence that x_1 and x_2 have some level of linear dependency, at least for this sample. The corresponding principal components z_1 and z_2 and their axes are also displayed. It can also be observed that the dispersion (or variance) is much more pronounced in the direction of z_1 than in z_2. Therefore, if we properly define $z_1 = \alpha x_1 + \beta x_2$ in the direction of the straight line that concentrate the dispersion of points, as shown in the figure, then the problem should be explained with only one variable z_1, that accounts for the most of the variance within the data.

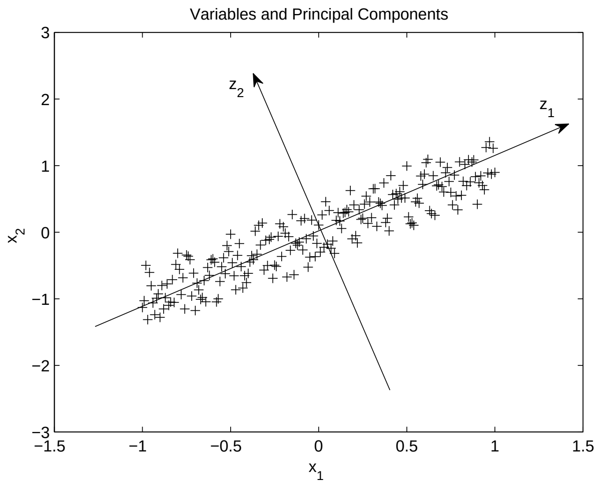


Figure 63: Geometric interpretation of variables and the respective principal components in two dimensions.

By the definition of principal components, it can be shown that the respective variance is given by

$$\mathrm{Var}\,(z_i) = \mathrm{Var}\left(\mathbf{V}_k^T\mathbf{x}\right) = \lambda_i, \tag{84}$$

and, as we explained, the higher the variance of a PC, the higher is its capacity to *explain* the data.

In the sequence, let us assume that the principal components are arranged in order of decreasing variance, i.e., ($\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_p > 0$). The idea is to remove the r PC with smallest variances, i.e., nearly zero eigenvalues, and to apply the least squares estimate to the remaining $(p-r)$ components, that is

$$\hat{\boldsymbol{\gamma}}_{PC} = \left[\hat{\gamma}_1 \;\; \hat{\gamma}_2 \;\; \cdots \;\; \hat{\gamma}_{p-r} \;\; | \;\; 0 \;\; 0 \;\; \cdots \;\; 0\right]^T . \tag{85}$$

Then, by applying the orthogonal transformation backwards, in order to get the original coefficients, it follows that

$$\boldsymbol{\beta}_{PC} = \mathbf{V}\hat{\boldsymbol{\gamma}}_{PC} = \sum_{j=1}^{p-r} \lambda_j^{-1} \mathbf{V}_j^T \mathbf{X}^T \mathbf{y} \mathbf{V}_j \ . \tag{86}$$

It is worth noting that, by removing less significant principal components z_i not necessarily delete variables x_i from the original model, since each PC is a linear combination of all x_i. On the other hand it is observed that by using small subset of PC offers considerable improvement over standard least squares when the data are ill-conditioned and the original regressors have linear dependencies, as we shown previously in the example of Figure 63.

Jolliffe (2002) presents a comprehensive review on the methods developed to choose a subset of principal components and variables. The immediate approach for choosing the $(s = p - r)$ PC to be kept in the model is based on the cumulative percent of total variation. In this sense it is desired to retain the selected PC that contribute with, say 80% to 90% of total variation. By defining the ratio

$$t_s = 100\% \times \frac{\sum\limits_{k=1}^{s} \lambda_k}{\sum\limits_{k=1}^{p} \lambda_k} \ , \tag{87}$$

and choosing a threshold, i.e., t^*, somewhere in the range 70% to 90% and retaining the least s PC such as ($t_s > t^*$), a rule, which in practice preserves in the first s PC most of the information in $\mathbf{x}$, is provided.

Another possibility is to normalize the eigenvalues, for example

$$\bar{\lambda}_i = \frac{\lambda_i}{\max\left(\lambda_i\right)} \ , \tag{88}$$

and to remove the PC proportionally smaller eigenvalues. For example, by setting a target value for instance $\bar{\lambda}^* > 10\%$, then all PC with normalized eigenvalues $\left(\bar{\lambda}_i < \bar{\lambda}^*\right)$ will be removed from the selected set and, in this way, it is possible to mitigate most of the ill-conditioning present in $\mathbf{X}^T\mathbf{X}$.

It is observed that targets for removing eigenvalues like t^*, $\bar{\lambda}^*$ or other variants are subjective and problem dependent. Therefore, it is recommended to apply some kind of iterative approach to select the correct cut-off value for each problem at hand.

Golub et al (1979) suggested a modified version of GCV statistic in order to handle variable selection for principal components, i.e.,

$$\mathrm{GCV}\left(s\right) = \frac{\frac{1}{N}\left\|\mathbf{I}-\mathbf{A}\left(s\right)\right\|^2}{\left[\frac{1}{N}\mathrm{tr}\left(\mathbf{I}-\mathbf{A}\left(s\right)\right)\right]^2} \ , \tag{89}$$

where the matrix $\mathbf{A}\left(s\right)$ is given by

$$\mathbf{A}\left(s\right) = \mathbf{UD}\left(s\right)\left[\mathbf{D}\left(s\right)^T\mathbf{D}\left(s\right)\right]^{-1}\mathbf{D}\left(s\right)^T\mathbf{U}^T \ , \tag{90}$$

in which $\mathbf{D}\left(s\right)$ is a diagonal matrix calculated from the *singular value decomposition* of $\mathbf{X}$, i.e.,

$$\mathbf{X} = \mathbf{UDV}^T \ , \tag{91}$$

by setting all but the s-th subset of singular values equal to 0, and, in the same way, the optimal s is selected to be $s = \arg\min_s \mathrm{GCV}\left(s\right)$.

It can be found in the literature many other variants of linear regression based on principal components or with close connections to it with the objective to produce biased estimates to remove colinearities and/or improve the accuracy of the fit. For example *latent root regression* Midi and Hua (2009) and *partial least squares*, PLS, Garthwaite (1994). However, based on the discussion driven by Jolliffe (2002), there is no evidence that any of these methods are absolutely superior to others in the comparative studies published.

F.5.4 Penalized Least Squares: A Unified Approach

As we explained and discussed in the last sections, a key issue in least squares regression is to control the size, i.e., the norm of the least squares estimator vector $\hat{\boldsymbol{\beta}}$, in order to balance bias an variance and to find a more stable and accurate model.

Common constraints are added to the ordinary least squares problem, for example the *sum to unit* and/or the *positivity* of the coefficients, i.e.,

$$\sum_{i=1}^{p}\beta_i = 1 \qquad \text{and} \qquad \beta_i \geq 0 \ .$$

In these cases the problem now is known as *constrained* least squares regression.

By using a slightly different approach, the least squares problem can be stated as an optimization problem, where the objective is to find $\hat{\boldsymbol{\beta}}$ that minimizes a suitable function of $\|\boldsymbol{\beta}\|$, under the constraint $\mathbf{y} = \mathbf{X}\boldsymbol{\beta}$, for instance, by following the derivation proposed in Fang et al (2006),

$$\|\boldsymbol{\beta}\|^2 + \lambda_0 \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2 \ , \tag{92}$$

where $\|\cdot\|$ is the Euclidean norm and λ_0 is a Lagrange multiplier, and by considering this problem as a *penalized sum of least squares*

$$\frac{1}{2}\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2 + \lambda \|\boldsymbol{\beta}\|^2 \ , \tag{93}$$

where λ now is referred to as a *regularization parameter* or *tunning parameter*. The solution of (93) can be expressed as

$$\hat{\boldsymbol{\beta}}_\lambda = \left(\mathbf{X}^T\mathbf{X} + 2\lambda \mathbf{I}_p\right)^{-1} \mathbf{X}^T \mathbf{y} \ , \tag{94}$$

where $\mathbf{I}_p$ is the identity matrix of order p. As can be noted, this solution is nothing but the ridge least squares estimate, that we presented and discussed previously.

Therefore the concept of *penalized least squares* allows to extend the concept of variable and model selection, since variable selection can be stated as a type of regularization problem.

By considering other penalty functions, let us define a general *penalized least squares* as

$$Q(\boldsymbol{\beta}) = \frac{1}{2}\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2 + N\sum_{i=1}^{p} \mathcal{P}_\lambda(|\beta_i|) \ , \tag{95}$$

where $\mathcal{P}_\lambda(\cdot)$ is a pre-specified nonnegative penalty function, and λ is a regularization parameter, which may depend on N and can be chosen by any data dependent criterion such as cross-validation. The penalized least squares minimization problem leads to a *penalized least squares estimator*. The regularization parameter λ controls the level of bias and variance and, eventually the complexity of the model, i.e., when λ increases the model tends to be simpler (more parsimonious) and the opposite is also true.

Many traditional variable selection methods can be derived in the form of penalized least squares, by choosing suitable penalty function $\mathcal{P}_\lambda(\cdot)$ and the regularization parameter λ. For instance, by using the L_0 penalty

$$\mathcal{P}_\lambda(|\beta_i|) = \frac{1}{2}\lambda^2 I(|\beta_i| \neq 0) \ , \tag{96}$$

where $I(\cdot)$ is the *indicator function*, i.e.,

$$I(\xi): \begin{cases} 1 \ \text{ if } \ \xi \in \chi \\ 0, \text{otherwise.} \end{cases} . \tag{97}$$

If we consider for example the variable selection methods cited in Section F.4, i.e., Mallows C_p AIC, BIC, ϕ -criterion and RIC, and apply the general form in Equation (96) it is possible to define specific formulas for λ, as function of the variation σ, the number of samples N and/or the number of selected coefficients s. See Fan and Li (2001) and Fang et al (2006) for a good account on these methods and their connections to the penalized least squares.

By using other penalty forms, different methods can be proposed, for instance, the L_q penalty, as proposed by Frank and Friedman (1993), also known as *bridge regression*

$$\mathcal{P}_\lambda\left(|\beta_i|\right) = \lambda\,|\beta_i|^q \ , \qquad q > 0, \tag{98}$$

and the so called LASSO (least absolute shrinkage and selection) algorithm, by Tibshirani (1996), which is based on the L_1 penalty in the form

$$\mathcal{P}_\lambda\left(|\beta_i|\right) = \lambda\,|\beta_i|\,. \tag{99}$$

As detailed by Breiman (1996), variable selection methods in regression suffer from several drawbacks, like computational cost and lack of stability. Motivated to drive these issues, Fan and Li (2001) proposed the SCAD (smoothly clipped absolute deviation) penalty and compared its performance with other methods for variable selection. They advocate that SCAD is an improvement by saving computational cost and resulting in a continuous solution to avoid unnecessary modeling variation and excessive introduction of bias.

The SCAD penalty is then defined as follows. Suppose that the initial value $\boldsymbol{\beta}_0$ obtained from ordinary least squares solution is close to the minimizer of Equation (95). If β_{i0} is close to zero, then set $\beta_{i0} = 0$ and the penalty function is estimated in the neighborhood of $\boldsymbol{\beta}_0$ (i.e., by Taylor's series expansion) as

$$\mathcal{P}_\lambda\left(|\beta_i|\right) \approx \mathcal{P}_\lambda\left(|\beta_{i0}|\right) + \frac{1}{2}\left\{\frac{\mathcal{P}'_\lambda\left(|\beta_{i0}|\right)}{|\beta_{i0}|}\right\}\left(\beta_i^2 - \beta_{i0}^2\right) \ , \qquad \text{for} \qquad \beta_i \approx \beta_{i0} \ , \tag{100}$$

with the first derivative of penalty function given by

$$\mathcal{P}'_\lambda\left(\beta_i\right) = \lambda\left\{I\left(\beta_i \leq \lambda\right) + \frac{\left(a\lambda - \beta_i\right)_+}{\left(a-1\right)\lambda} I\left(\beta_i > \lambda\right)\right\} \ , \tag{101}$$

for some $a > 2$ and $\beta_i > 0$ and $I\left(\cdot\right)$ the indicator function.

In this way, $\mathcal{P}_\lambda\left(|\beta_i|\right)$ is defined continuous and differentiable up to second order, a property that provide advantages over L_0 and L_1 penalties, for example, in terms of stability, moderate bias and computational cost, as discussed in Fan and Li (2001). The final solution for least squares by SCAD penalty is driven by an iterative process similar to ridge regression. See details of this algorithm in Fan and Li (2001) or Fang et al (2006). The convergence properties of this algorithm was studied and presented in Hunter and Li (2005).

In Fang et al (2006) it is presented a detailed explanation on penalized least squares methods like, SCAD, LASSO in comparison to stepwise regression by F-statistic (as in Efroymson algorithm), AIC, BIC RIC and ϕ-criterion. In one application example in regression with 6 variables, presented in Fang et al (2006) (pp. 259-261), surprisingly all the other implemented variable selection yielded exactly the same regression coefficients except LASSO. In this example, LASSO slightly shrunk the regression coefficients and yielded a slightly larger sum of squared errors.

Finally, variable selection methods is a large front of research in least squares approximation field. Miller (2002) presented an extensive review on variable selection in regression problems and this is still a subject of active research, as can be seen in the recent publications, for instance Ng (2012) which states that: "The variable selection is by no means solved." and Scheipl et al (2013) that reinforces that there is still a wide and open field for future research in variable and function selection in multivariate regression.

F.6 Other Least Squares Variants to Improve Accuracy

F.6.1 Effect of the Intercept Term

Let us recall the definition of linear ensemble, i.e.,

$$\hat{y}_{ens}\left(w_i\ ,\mathbf{x}\right)=w_0+\sum_{i=1}^{M}w_i\hat{y}_i\left(\mathbf{x}\right). \tag{102}$$

The majority of methods does not take into account the intercept term, or the constant w_0 for generating the linear ensemble of models, i.e., $w_0 = 0$.

In the study developed by Hashem (1993), it was remarked the effect of the intercept in the accuracy of the final ensemble of neural networks. They identified that for well-trained network models, the optimal sum of weights tends to one and the constant term tends to zero. However, for poorly trained networks, the sum of optimal weights is far from one and the constant term is significantly different from zero. The authors noted that these results evidenced the role of ensemble in the approximation in case of poorly trained networks and, on the contrary, the only "fine-tuning" role of the ensemble in case of well trained networks.

In regression analysis this effect of intercept is also known, as can be viewed in Figure 64. As discussed by Montgomery et al (2006), the effect can be significant specially when data lie in a region far from the origin. In many cases a model with intercept provides a much better fit in the region of space where the data were collected.

Therefore there is no reason *a priori* to neglect the intercept term in the ensemble models. Since it is generally fast to generate the linear ensemble, it can be a good practice to check whether the intercept term improves the accuracy of approximation or not.

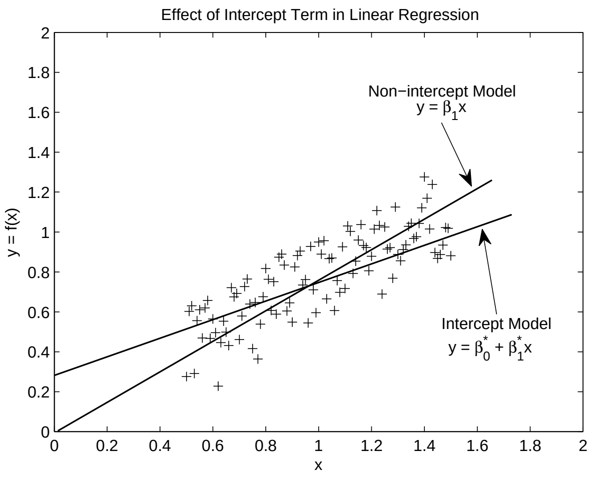


Figure 64: Difference of considering or neglecting the intercept term in regression for one dimensional problem.

F.6.2 Generalized Least Squares

In the linear regression equation, i.e.,

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon} \ , \tag{103}$$

the basic assumption is that the random errors are normally and independently distributed (NID), with zero mean and finite variance, i.e., $\varepsilon_i \in$ NID$(0, \sigma^2)$, which means, in other words, that the errors are *uncorrelated* and have *constant variance*, or

$$\mathrm{Var}\left(\varepsilon\right) = \sigma^2 \mathbf{I} \ , \tag{104}$$

where $\mathbf{I}$ is the identity matrix of order N.

In many cases these assumptions do not hold and the observations $\mathbf{y}$ should be *correlated* and have *unequal variances*, i.e.,

$$\mathrm{Var}\left(\varepsilon\right) = \sigma^2 \boldsymbol{\Sigma} \ , \tag{105}$$

where $\boldsymbol{\Sigma}$ is a general $(N \times N)$ matrix , the *variance-covariance matrix*, with distinct diagonal terms and off-diagonal terms nonzero.

In situations like that, the least squares estimator $\hat{\boldsymbol{\beta}} = \left(\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{X}^T\mathbf{y}$ is not as accurate as expected and assured by Gauss-Markov theorem. Therefore some modifications to least squares solution are necessary in order to make the estimation reasonable.

Let us assume that it is possible to define a linear matrix transformation, as follows

$$\mathcal{G}\mathbf{y} = \mathcal{G}\mathbf{X}\boldsymbol{\beta} + \mathcal{G}\boldsymbol{\varepsilon} \ . \tag{106}$$

As direct consequence, we have

$$\hat{\boldsymbol{\beta}}_{\mathcal{G}} = \left(\mathbf{X}^T\mathcal{G}^T\mathcal{G}\mathbf{X}\right)^{-1}\mathbf{X}^T\mathcal{G}^T\mathcal{G}\mathbf{y} \qquad \text{and} \qquad \operatorname{Var}\left(\mathcal{G}\varepsilon\right) = \sigma^2\mathcal{G}\boldsymbol{\Sigma}\mathcal{G}^T \ . \tag{107}$$

In order to make the error assumptions true, we need to force

$$\operatorname{Var}\left(\mathcal{G}\boldsymbol{\varepsilon}\right) = \sigma^2\mathcal{G}\boldsymbol{\Sigma}\mathcal{G}^T = \sigma^2\mathbf{I}. \tag{108}$$

Then, since $\boldsymbol{\Sigma}$ is symmetric and positive definite, by the variance-covariance definition, thus it can be diagonalized as $\boldsymbol{\Sigma} = \mathbf{S}\boldsymbol{\Lambda}\mathbf{S}^T$, where $\mathbf{S}$ and $\boldsymbol{\Lambda}$ are the corresponding matrices of eigenvectors and eigenvalues of $\boldsymbol{\Sigma}$.

Since $\boldsymbol{\Sigma}^{-\frac{1}{2}} = \mathbf{S}\boldsymbol{\Lambda}^{-\frac{1}{2}}\mathbf{S}^T$, then $\boldsymbol{\Sigma}^{-\frac{1}{2}}\boldsymbol{\Sigma}\left(\boldsymbol{\Sigma}^{-\frac{1}{2}}\right)^T = \mathbf{I}$, therefore $\mathcal{G}$ must be proportional to $\boldsymbol{\Sigma}^{-\frac{1}{2}}$, i.e.,

$$\mathcal{G} = \nu\boldsymbol{\Sigma}^{-\frac{1}{2}} \ , \tag{109}$$

for some constant ν.

Now, for this choice of $\mathcal{G}$, the assumptions on the error variance hold and the *generalized least squares estimator* of $\boldsymbol{\beta}$ is given by

$$\hat{\boldsymbol{\beta}}_{gls} = \left(\mathbf{X}^T\boldsymbol{\Sigma}^{-1}\mathbf{X}\right)^{-1}\mathbf{X}^T\boldsymbol{\Sigma}^{-1}\mathbf{y} \qquad \text{with} \qquad \operatorname{Var}\left(\hat{\boldsymbol{\beta}}_{gls}\right) = \sigma^2\left(\mathbf{X}^T\boldsymbol{\Sigma}^{-1}\mathbf{X}\right)^{-1} \ . \tag{110}$$

In practice, the problem is on how to define the covariance matrix $\boldsymbol{\Sigma}$. In general, this matrix is not known *a priori*, or even in some cases $\boldsymbol{\Sigma}$ is found to be singular. Therefore it must be estimated based on convenient assumptions regarding the errors $\boldsymbol{\varepsilon}$ shape or structure on their variances. In this sense, when $\boldsymbol{\Sigma}$ is estimated, the method is known as *feasible generalized least squares.*

This kind of problem commonly arises in time series analysis in econometrics, for example. Further details on generalized least squares theory and methods to estimate $\boldsymbol{\Sigma}$ can be found in Kuan (2012) and Amemiya (1985).

In general, the solution for generalized least squares problems is carried on with an iterative process. The first step is the ordinary least squares and in the sequence $\boldsymbol{\Sigma}$ is estimated in order to get the feasible generalized least squares estimator $\hat{\boldsymbol{\beta}}_{fgls}$. The method is repeated until some convergence criterion is fulfilled, for example, at the iteration $(k+1)$

$$\max_i \frac{\left|\hat{\beta}_{i,(k+1)} - \hat{\beta}_{i,(k)}\right|}{\left|\hat{\beta}_{i,(k)}\right|} < \xi \ , \tag{111}$$

for the i-th component of $\hat{\boldsymbol{\beta}}$ and ξ is the tolerance, which in general is set to 10^{-8}.

In the related bibliography, Amemiya (1985), it is possible to find different approaches to estimate $\boldsymbol{\Sigma}$, based on the correlation pattern assumed for the errors. The multicollinearity problem in least squares is mainly related to linear dependencies on the *columns* of the matrix $\mathbf{X}$, i.e., some level of redundancy in the variables. On the other hand, correlation in the errors is in general associated to linear dependencies on the *lines* of $\mathbf{X}$, i.e., some redundancy in the observations of y_i. The redundancy in the observations is common to arise in problems where the data are collected repeatedly across the time, which is proper to time series analysis in Economics, for instance.

In other contexts, the correlation of y_i or ε_i is not existent or it is not severe and thus it can be neglected. That is our case, since the data is generated by designed experiments methods (DOE) and the uniqueness of the lines of $\mathbf{X}$ is in general forced. On the other hand, we have no control on the variance of ε_i and it is not possible to assure that it will be constant for each line i.

In cases like that, where is reasonable to assume uncorrelation of ε_i and non constant variance, it is possible to assume $\mathbf{\Sigma}^{-1}$ as a diagonal matrix $\mathbf{W}$, with distinct terms on the diagonal. This method is known as *weighted least squares*, since each line is weighted by it respective variance, then the least squares estimation is

$$\hat{\boldsymbol{\beta}}_{wls} = \left(\mathbf{X}^T\mathbf{W}\mathbf{X}\right)^{-1}\mathbf{X}^T\mathbf{W}\mathbf{y} \qquad \text{with} \qquad \mathrm{Var}\left(\hat{\boldsymbol{\beta}}_{wls}\right) = \sigma^2\left(\mathbf{X}^T\mathbf{W}\mathbf{X}\right)^{-1} , \tag{112}$$

where $\mathbf{W}$ is given by

$$\mathbf{W} = \begin{bmatrix} w_{11} & & & 0 \\ & w_{22} & & \\ & & \ddots & \\ 0 & & & w_{NN} \end{bmatrix} , \tag{113}$$

and thus,

$$\mathrm{Var}\left(\varepsilon\right) = \sigma^2\mathbf{W}^{-1} . \tag{114}$$

Now, the question is centered on the estimation of the N weights w_{ii}. See for example Weisberg (1985) for a discussion on different methods for this purpose.

One feasible approach is to regress the residuals $e_i = y_i - \hat{y}_i$, in terms of the variables x_i,

$$\mathbf{e} = \mathbf{X}\boldsymbol{\beta}_{\mathbf{e}}, \tag{115}$$

and then estimate $\mathrm{Var}\left(\mathbf{e}\right)$ by

$$\mathrm{Var}\left(\hat{e}_i\right) = \sigma_e\mathbf{x}_i^T\left(\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{x}_i, \tag{116}$$

with σ_e estimated by

$$\hat{\sigma}_e = \frac{\mathbf{e}^T\mathbf{e} - \hat{\boldsymbol{\beta}}_{\mathbf{e}}^T\mathbf{X}^T\mathbf{e}}{N-p} . \tag{117}$$

And finally, $\mathbf{W}$ is estimated by

$$\mathbf{W} = \begin{bmatrix} \frac{1}{\hat{\Sigma}_{11}} & & & 0 \\ & \frac{1}{\hat{\Sigma}_{22}} & & \\ & & \ddots & \\ 0 & & & \frac{1}{\hat{\Sigma}_{NN}} \end{bmatrix} , \tag{118}$$

with $\hat{\Sigma}_{ii} = \mathrm{Var}\left(\hat{e}_i\right)$.

The most widely used procedure is the *iteratively reweighted least squares*, IRLS. The starting point is to perform an ordinary least squares and estimate the coefficients $\hat{\boldsymbol{\beta}}_0$ and the weighing matrix $\mathbf{W}_0$. Then process iterates by reestimating $\boldsymbol{\beta}_k$ and $\mathbf{W}_k$, i.e.,

$$\hat{\boldsymbol{\beta}}_{k+1} = \left(\mathbf{X}^T\mathbf{W}_k\mathbf{X}\right)^{-1}\mathbf{X}^T\mathbf{W}_k\mathbf{y} \ , \tag{119}$$

until some convergence is achieved, that can be based on Equation (111), i.e., the percent variation of coefficients $\hat{\boldsymbol{\beta}}_{k+1}$, with respect to the previous step, i.e., $\hat{\boldsymbol{\beta}}_k$.

In this way, after the convergence of the iterative process, the weighted least squares solution $\hat{\boldsymbol{\beta}}_{wls}$ is expected to be more accurate than the ordinary least squares $\hat{\boldsymbol{\beta}}_0$, when the errors are not correlated and their variances are not constant.

F.6.3 L_p-Norms and Robust Regression

When the assumption of normally distributed output $\mathbf{y}$ in the linear regression model holds, the method of least squares provides estimates with good statistical properties. However, there are many situations where this assumption is not realistic and the distribution of $\mathbf{y}$ is clearly nonnormal and/or there are data points (i.e., outliers) that affect the final regression model accuracy.

In physical measurements, the outliers should be spurious data, and, whenever is possible, they are removed from the sampling space. On the other hand, there are several situations that these points cannot be considered as outliers, because they are part of the deterministic problem, as for instance in highly nonlinear phenomena, like in vehicle crash simulations or CFD analyses. In these cases, other methods or variants of least squares solutions, less sensitive (or more *robust*) to the presence of outliers, must be applied.

In order to make an illustration, let us see the response pattern of some numerical benchmark and engineering functions, by means of *normal probability plots*. The graphs in Figure 65 present the sampling distribution of the response $\mathbf{y}$ of benchmark functions from $n_v = 2$

to $n_v = 10$ variables, and the number of points in the sampling space are set in two levels, $N = 10n_v$ (graphs on the left) and $N = 100n_v$ (graphs on the right). All the points have been generated by using optimized latin hypercube sampling by means of the the Matlab function `lhsdesign`, with `maxmin` criterion and `iterations` = 1000. The details of the benchmark functions can be found in the Appendix D.

When the distribution can be considered normal, the data in the normal plot should lie in a straight line or a nearly straight line in the range of interest. It can be observed that the normality assumption is not reasonable for most of the functions displayed, with moderate sampling points density, i.e., $N = 10n_v$, which is a usual value for metamodeling purposes in engineering applications.

Even in extremely high densities, $N = 100n_v$ (which is not usual or should be unfeasible in practice), the normal distribution is not realistic in the whole data ranges. In addition, this behavior is not dependent on the number of variables but on the problem itself, as can be observed for instance in the response for Nowacki Bending (2 variables), Figs. 65 (e) and (f), with a nonnormal distribution and on the contrary, the LiftSurf Weight response (10 variables), Figs. 65 (g) and (h), with a nearly normal distribution.

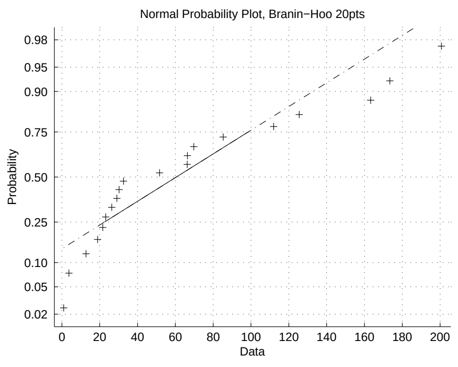
By using *robust* regression methods, the nonnormality issue can be handled. A *robust* procedure is one that make the least squares solution insensitive or less sensitive to outliers or the so called "heavy-tailed" distributions, such as the ones shown in Figure 65. In addition to insensitivity to this kind of distribution or outliers, a robust estimation procedure should provide the same results as ordinary least squares when the data range is normal and there are no "outliers".

The literature regarding robust regression is vast and a lot of classes of robust procedures have been proposed. See for example the books by Huber and Rochetti (2009) and Rousseeuw and Leroy (2003). In this work we will follow the derivation for robust regression presented in Montgomery et al (2006), specifically the class of robust methods known as *M-Estimators.*

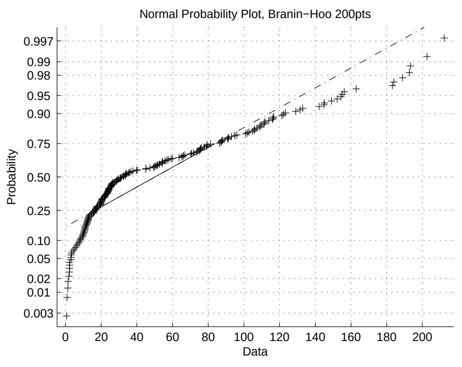
The first regression method developed with robustness property is the *least absolute deviation*, also known as *least modulo.* As described by Björk (1996), there is a historical controversy regarding the invention of least squares methods. It is attributed to Laplace in 1799, the regression by the principle of minimizing the the sum of absolute errors, i.e., $\sum_{i=1}^{N} |e_i|$, or L_1-norm. The algebraic principle of what is known as least squares i.e., $\sum_{i=1}^{N} e_i^2$ or L_2-norm, was first published by Legendre in 1805, and Gauss claimed that himself devised the statistical analysis of least squares as a better solution than least absolute errors in 1795.

According to Montgomery et al (2006), in 1887 F. Y. Edgeworth argued that least squares was overly influenced by large outliers, issue that could be solved by using L_1-norm. In this sense, the L_1-norm regression is thus a special case of the family of L_p-norm regression, in which the model parameters are chosen to minimize $\sum_{i=1}^{N} |e_i|^p$, where $1 \leq p \leq 2$, and the problem can be formulated and solved by means of nonlinear programing techniques.

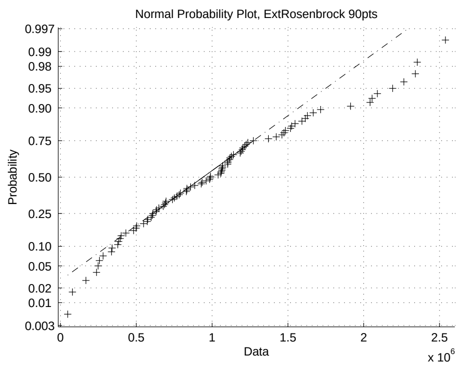
In the cases that normal distribution is not a valid assumption, it can be reasonable to model the errors by another convenient distribution. Of course the distribution pattern is not known in advance, but after running an ordinary least squares, for example, it is possible to infer the error distribution by normal probability plots or another convenient approach and then proceed to refining the approximation iteratively. The L_1-norm regression problems arises naturally from the *maximum-likelihood* approach with *double-exponential error distribution* modeling, as described in Montgomery et al (2006).



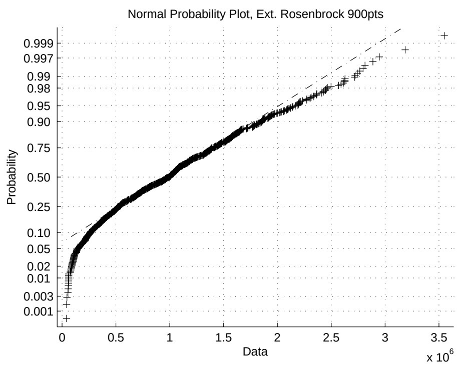
(a) Branin-Hoo, $n_v = 2$, $N = 20$.



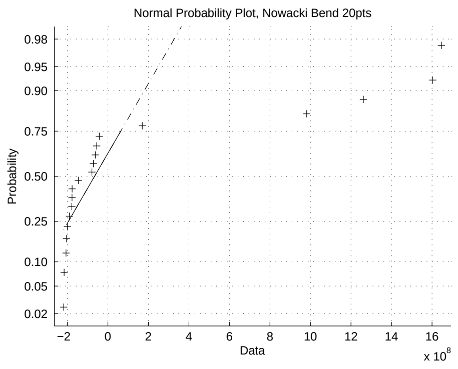
(b) Branin-Hoo , $n_v = 2$, $N = 200$.



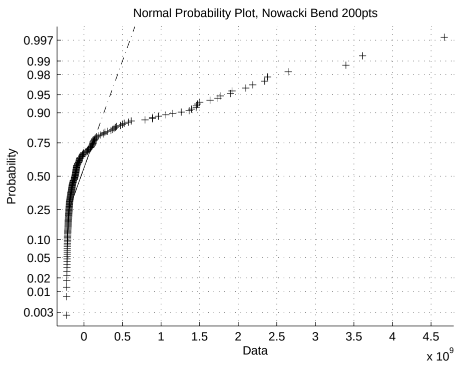
(c) Ext. Rosenbrock, $n_v = 9$, $N = 90$.



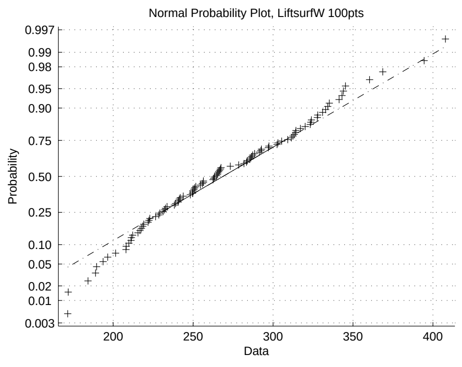
(d) Ext. Rosenbrock, $n_v = 9$, $N = 900$.



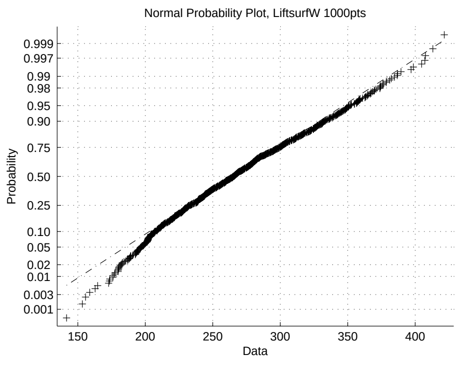
(e) Nowacki Bending, $n_v = 2$, $N = 20$.



(f) Nowacki Bending, $n_v = 2$, $N = 200$.



(g) Liftsurf W, $n_v = 10$, $N = 100$.



(h) Liftsurf W, $n_v = 10$, $N = 1000$.

Figure 65: Normal Probability plots for numerical benchmarks and engineering functions with different number of variables, n_v and number of sampling points, N. Straight line indicate perfect normal distribution of data.

In this way, it is possible to define a class of *robust estimators* that minimize a function $\rho\left(e_i\right)$, for example,

$$\min_{\boldsymbol{\beta}} \sum_{i=1}^{N} \rho\left(e_i\right) = \min_{\boldsymbol{\beta}} \sum_{i=1}^{N} \rho\left(\frac{y_i - \mathbf{x}_i^T \boldsymbol{\beta}}{s}\right) , \tag{120}$$

where $\mathbf{x}_i$ denotes the i-th row of $\mathbf{X}$ and s is a tuning parameter to properly scale the errors. A common choice of for s is

$$s = \frac{1}{0.6745} \text{median}(|e_i - \text{median}\left(e_i\right)|) , \tag{121}$$

that leads to unbiased estimator of the variance σ for large number of samples N.

This kind of estimator is called *M-Estimator*, where M stands for *maximum-likelihood*. Therefore, with appropriate choices of $\rho\left(e_i\right)$ to model the error distribution, it is possible to find different estimators to model the problem by regression. If one set for instance, $\rho\left(z\right) = \frac{1}{2}z^2$, $-\infty < z < \infty$, and $s = 1$, the standard least squares estimator is achieved. Several other functions $\rho\left(z\right)$ can be defined, and the most popular are: *Huber's t*, *Hamsay E_a*, *Andrews' wave* and *Hampel's 17A*. The functions $\rho\left(z\right)$, also called *influence functions*, differ by the softness or severity when damping the effect of outliers in the least squares response. See Montgomery et al (2006) for details on these functions.

The minimization procedure in Eq. (120) can be handled by nonlinear optimization techniques, but the traditional procedure is the iteratively re-weighted least squares, IRLS, as in Equation (119), for the weighted least squares procedure. The starting point is to lead a ordinary least squares and estimate the coefficients $\hat{\boldsymbol{\beta}}_0$, the scale parameter s. Then the process iterates by re-estimating $\boldsymbol{\beta}_k$ and the matrix $\mathbf{W}_k$, until convergence is achieved. In matrix notation,

$$\hat{\boldsymbol{\beta}}_{k+1} = \left(\mathbf{X}^T \mathbf{W}_k \mathbf{X}\right)^{-1} \mathbf{X}^T \mathbf{W}_k \mathbf{y} , \tag{122}$$

where $\mathbf{W}_k$ is a $N \times N$ diagonal matrix of "robust weights" defined by

$$w_{ii(k)} = \begin{cases} \dfrac{\psi\left(\tilde{z}_{i(k)}\right)}{\tilde{z}_{i(k)}} & \text{if} \quad \tilde{z}_{i(k)} \neq 0 \\ 1 & \text{if} \quad \tilde{z}_{i(k)} = 0 \end{cases} , \tag{123}$$

where $\tilde{z}_{i(k)} = \dfrac{\left(y_i - \mathbf{x}_i^T \hat{\boldsymbol{\beta}}_k\right)}{s_k}$ and $\psi\left(z\right) = \dfrac{d\rho\left(z\right)}{dz}$. As stated by Montgomery et al (2006), usually only a few iterations are required to achieve convergence. Therefore, depending on the magnitude of the errors, represented by $\tilde{z}_i$, and the shape of the function $\psi\left(z\right)$, then different weights w_{ii} are applied. In other words, the higher the error, the lower the weight.

There is no general agreement about the estimation of the variance of the robust coefficients $\hat{\boldsymbol{\beta}}_{rob}$. There are many formulas for this purpose, based on different assumptions, but there is no consensus on which one is the best. One common estimate is

$$\mathrm{Var}\left(\hat{\boldsymbol{\beta}}_{rob}\right) = \frac{\sum\limits_{i=1}^{N} w_i \left(y_i - \mathbf{x}_i^T \hat{\boldsymbol{\beta}}\right)^2}{N-p} \left(\mathbf{X}^T \mathbf{W} \mathbf{X}\right)^{-1} . \tag{124}$$

In Figure 66 it is shown a hypothetical example by comparing standard least squares solution and robust regression. The 100 scatter points has been generated by adding random normal noise to the trend line ($y = 10 - 2x$). The outliers have been artificially generated by modifying the response of the last 5 points. Note that the standard least squares solution is affected by the presence of outliers, i.e., the LS regression line ($y = -16.52 - 1.19x$) is moved in the direction of the outliers, in order to balance the vertical distances, since the method considers equal weights for all the errors. On the other hand, by applying different weights $w_i i$ for the errors, the robust regression line ($y = 10.81 - 2.04x$) "ignores" the presence of outliers and it better approximates the "true trend line" of the data if we remove the outliers and apply standard least squares to the remaining points, i.e., ($y = 10.05 - 2.00x$).

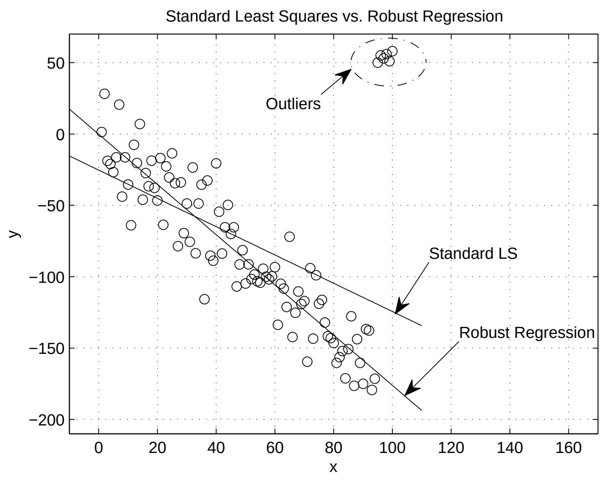


Figure 66: Comparison between least squares and robust regression.

As remarked by Montgomery et al (2006), in practical applications of regression the two more frequent problems encountered are nonnormality of the observations and multicollinearity. Although both issues are from different nature and sources, in a significant number of applications, nonnormal distributions (with or without outliers) and lack of linear independence of the variables occur simultaneously. Several authors have suggested that either robust and biased estimation methods should be sufficient for handling both issues at same time. In addition, it is worth noting that robust regression estimates are frequently unstable when $\mathbf{X}$ is ill-conditioned, thus it would be desirable to have a technique for solving both problems at same time.

If we rewrite the robust minimization, by adding a constraint as in ridge regression

$$\min_{\boldsymbol{\beta}} \sum_{i=1}^{N} \rho\left(\frac{y_i - \mathbf{x}_i^T\boldsymbol{\beta}}{s}\right) \quad , \qquad \text{subject to} \quad \boldsymbol{\beta}^T\boldsymbol{\beta} \leq d^2 \tag{125}$$

where the radius d depends on the ridge parameter k, then the optimum solution is a *ridge robust estimator* $\hat{\boldsymbol{\beta}}_{RR}$.

The IRLS algorithm can be modified by augmenting the matrix $\mathbf{X}$ and vector $\mathbf{y}$

$$\mathbf{X}_A = \begin{bmatrix} \mathbf{X} \\ \sqrt{k}\mathbf{I}_p \end{bmatrix}, \qquad \mathbf{y}_A = \begin{Bmatrix} \mathbf{y} \\ \mathbf{0}_p \end{Bmatrix}, \tag{126}$$

with $\mathbf{I}_p$ and $\mathbf{0}_p$ the identity matrix of order $p \times p$ and null vector of order $p \times 1$, respectively, and then, by changing the robust iteration procedure to estimate, for a certain ridge parameter k, i.e.,

$$\hat{\boldsymbol{\beta}}_{j+1} = \left(\mathbf{X}_A^T\mathbf{W}_j\mathbf{X}_A\right)^{-1}\mathbf{X}_A^T\mathbf{W}_j\mathbf{y}_A \ , \tag{127}$$

and then, by using this modified procedure, the ridge robust estimator $\hat{\boldsymbol{\beta}}_{RR}$ is obtained and it is possible to handle both multicollinearity and nonnormal data simultaneously.

F.6.4 Total Least Squares

The ordinary least squares method assume that the columns of the $\mathbf{X}$ matrix, i.e., the input variables, are error free and all the errors are confined to the right hand side term, i.e., the response vector $\mathbf{y}$ (the output variable). However, in many applications this assumption is not realistic, i.e., both $\mathbf{X}$ and $\mathbf{y}$ are contaminated by noise and the ordinary least squares solution is not accurate anymore. This kind of problem arises in several fields, for example: signal processing, modal and spectral analysis, linear system theory, system identification and astronomy. See the review Markovsky and van Huffel (2007) and the book van Huffel and Vandewalle (1991) for a good account on the areas and applications that this problem is common.

Motivated to solve this issue, 40 to 50 years ago, statisticians developed methods such as "orthogonal regression", "errors-in-variables models" and "measurement errors". In fact, it is attributed to R. Addock in 1877 the first development in this sense, *apud* Markovsky and van Huffel (2007). More recently, in the field of numerical analysis, this kind of problem has been studied and the solution formalized in terms of *singular value decomposition*, SVD, by Golub and van Loan (1980), where the term "total least squares" has been first used.

The method is traditionally generalized to the multivariate case, i.e., the output is a matrix $\mathbf{Y}$ of order $(N \times d)$, where N is the number of observation and d is the number of different response vectors and ($N > d$). For simplicity, let us concentrate first on the univariate case, i.e., $d = 1$. The problem now is that both $\mathbf{X}$ and $\mathbf{y}$ are prone to errors, i.e., the matrix $\Delta\mathbf{X}$ and the vector $\Delta\mathbf{y}$, respectively

$$(\mathbf{X} + \Delta\mathbf{X})\,\boldsymbol{\beta} = \mathbf{y} + \Delta\mathbf{y} + \boldsymbol{\varepsilon}, \tag{128}$$

and the classical total least squares solution looks for the minimal correction matrix $\Delta\mathbf{X}$ and vector $\Delta\mathbf{y}$, in the following optimization problem

$$\min_{\Delta\mathbf{X},\,\Delta\mathbf{y}} \left\|[\Delta\mathbf{X} \quad \Delta\mathbf{y}]\right\|_F \ , \qquad \text{subjected to} \qquad (\mathbf{X} + \Delta\mathbf{X})\,\boldsymbol{\beta} = \mathbf{y} + \Delta\mathbf{y} \ , \tag{129}$$

where $\|\mathbf{A}\|_F = \sqrt{\mathrm{tr}\left(\mathbf{A}^T\mathbf{A}\right)}$ is the *Frobenius* norm of a matrix A.

Thus, differently to ordinary least squares, that minimizes the Euclidean norm of the error in the response vector $\mathbf{y}$, both errors in the variables matrix $\mathbf{X}$ and vector $\mathbf{y}$ are minimized in the Frobenius norm sense, when total least squares is applied. The total least squares estimator $\hat{\boldsymbol{\beta}}_{tls}$ is, in this way, the solution for the optimally corrected system of equations

$$\hat{\mathbf{X}}\boldsymbol{\beta} = \hat{\mathbf{y}} \ , \tag{130}$$

with $\hat{\mathbf{X}} = (\mathbf{X} + \Delta\mathbf{X})$ and $\hat{\mathbf{y}} = (\mathbf{y} + \Delta\mathbf{y})$.

It is shown by Golub and van Loan (1980) that the solution of total least squares can be driven by singular value decomposition (SVD). By following the notation and derivation presented in van Huffel and Vandewalle (1987), let the SVD of the $N \times (p + 1)$ augmented matrix $\boldsymbol{\Xi} = [\mathbf{X} \quad \mathbf{y}]$ be given by

$$\boldsymbol{\Xi} = \hat{\mathbf{U}}\hat{\boldsymbol{\Sigma}}\hat{\mathbf{V}}^T \ , \tag{131}$$

with the definitions

$$\begin{cases} \hat{\mathbf{U}} = [\hat{\mathbf{u}}_1, \cdots, \hat{\mathbf{u}}_{p+1}] & \hat{\mathbf{u}}_i \text{ vector of order} \quad (N \times 1) \ , \\ \hat{\mathbf{V}} = [\hat{\mathbf{v}}_1, \cdots, \hat{\mathbf{v}}_{p+1}] & \hat{\mathbf{u}}_i \text{ vector of order} \quad (p+1) \times 1 \ , \\ \hat{\mathbf{U}}^T\hat{\mathbf{U}} = \hat{\mathbf{V}}^T\hat{\mathbf{V}} = \mathbf{I}_{p+1} & \hat{\mathbf{U}} \text{ and } \hat{\mathbf{V}} \text{ are unitary matrices} \ , \\ \hat{\boldsymbol{\Sigma}} = \mathrm{diag}\left(\hat{\sigma}_1, \cdots, \hat{\sigma}_{p+1}\right) \quad \text{with} \quad \hat{\sigma}_1 \geq \cdots \geq \hat{\sigma}_{p+1} \ , \end{cases} \tag{132}$$

where $\hat{\mathbf{u}}_i$ and $\hat{\mathbf{v}}_i$ are respectively the *right* and *left* singular vectors and and $\hat{\sigma}_i$ the singular values of the augmented matrix $\boldsymbol{\Xi}$.

It is shown that, if $\hat{\sigma}_p > \hat{\sigma}_{p+1}$ (unique singular values) and $\mathbf{y}$ is not orthogonal to $\hat{\mathbf{u}}_{p+1}$, then the total least squares estimator is

$$\hat{\boldsymbol{\beta}}_{tls} = \left(\mathbf{X}^T\mathbf{X} - \hat{\sigma}_{p+1}^2\mathbf{I}\right)^{-1}\mathbf{X}^T\mathbf{y} \ , \tag{133}$$

or, by using the singular components (values and vectors) of $\boldsymbol{\Xi}$,

$$\hat{\boldsymbol{\beta}}_{tls} = \sum_{j=1}^{p} \left(\hat{\sigma}_j^2 - \hat{\sigma}_{p+1}^2\right)^{-1} \hat{\sigma}_j \left(\hat{\mathbf{u}}_i^T\mathbf{y}\right) \hat{\mathbf{v}}_j \ . \tag{134}$$

If $\hat{\sigma}_{p+1}$ coincides with other singular values of Ξ, the solution is a linear combination of the corresponding right singular values, such that $\hat{\boldsymbol{\beta}}_{tls}$ has minimum norm in the Frobenius sense. See van Huffel and Vandewalle (1991) for proofs and details.

In Figure 67, it is presented a geometrical interpretation of least squares and total least squares solution for the same data points in one dimension. The least squares (LS) try to find the minimum sum of squared vertical distances to the regression line (errors in y), while total least squares (TLS) try to minimize the sum of squared orthogonal distances (errors in both y and x) and, as a consequence, the solutions (regression lines) are different.

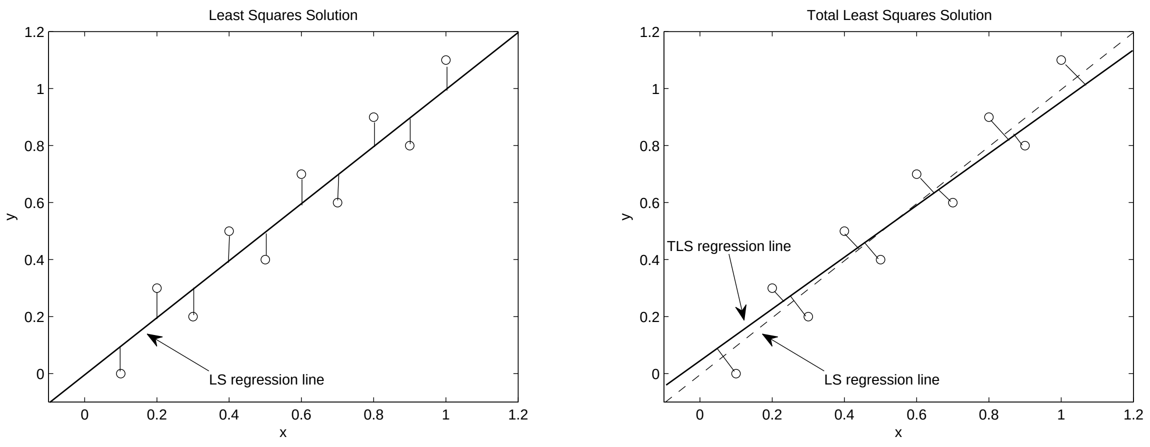


Figure 67: Comparison between least squares and total least squares solution.

The relationship of total least squares estimator and classical solutions, i.e., ordinary, ridge and principal components regression is presented in van Huffel and Vandewalle (1987). For instance, if we compare TLS and ridge, by means of Equations (133) and (67), if $k = -\hat{\sigma}^2_{p+1}$, then the total least squares solution can be interpreted as a *deregularizing* solution or a kind of *reverse* ridge regression. In this sense, the total least squares works by removing bias from the estimation, by subtracting the $\hat{\sigma}^2_{p+1}$ from $\mathbf{X}^T\mathbf{X}$, when both the input and output variables are subjected to errors.

In terms of variance, it is shown by van Huffel and Vandewalle (1991), that for independent and identically distributed errors,

$$\operatorname{Var}\left(\hat{\boldsymbol{\beta}}_{tls}\right) \approx \frac{\hat{\sigma}_{p+1}}{N}\left(1 + \left\|\hat{\boldsymbol{\beta}}_{tls}\right\|^2\right)\left(\mathbf{X}^T\mathbf{X} - N\hat{\sigma}^2_{p+1}\mathbf{I}\right)^{-1} \ . \tag{135}$$

Thus, the variance for total least squares estimators is larger than ordinary least squares. However, in terms of mean squared error (i.e., variance and bias balanced) the bias reduction and variance increasing can annihilate each other, producing comparable MSE for total least squares and ordinary least squares, for moderate sample sizes N. In case of increasing both noise level and sample sizes N, it is observed in simulation tests that total least squares solution becomes asymptotically superior in terms of accuracy. On the other hand, since the variance is in general higher in total least squares, it can lead to instabilities and lack of robustness in presence of outliers in the data.

Therefore, total least squares methods has emerged as a successful method for noise reduction in linear least squares problems in a number of applications. In a similar way to ordinary least squares methods, a lot of research have been conducted to improve the qualities and reduce the advantages of total least squares solutions. In this sense, variations in terms of regularization to handle multicollinearity, non contestant variances and robust methods, for instance, have been developed for total least squires as well. Additional connections, extensions, and sensitivity properties of total least squares and classical solutions are described in van Huffel and Vandewalle (1991). Recent developments and variants are discussed in Markovsky and van Huffel (2007).

The basic algorithm to solve classic total least squares, for the multivariate case, $d > 1$, can be depicted as follows, according to Fierro and Bunch (1997):

1. Compute the SVD of the augmented matrix $\mathbf{\Xi} = [\mathbf{X} \quad \mathbf{Y}]$ and consider the partitioning

$$\hat{\mathbf{V}} = \begin{bmatrix} \hat{\mathbf{V}}_{11}^{(p\times p)} & \hat{\mathbf{V}}_{12}^{(p\times d)} \\ \hat{\mathbf{V}}_{21}^{(d\times p)} & \hat{\mathbf{V}}_{22}^{(d\times d)} \end{bmatrix}; \tag{136}$$

2. A total least squares solution exists and it is unique if and only if $\hat{\mathbf{V}}_{22}$ is non-singular and $\hat{\sigma}_p \neq \hat{\sigma}_{p+1}$. In this case, the total least squares estimator is

$$\hat{\boldsymbol{\beta}}_{tls}^{*} = -\hat{\mathbf{V}}_{12}\left(\hat{\mathbf{V}}_{22}\right)^{\dagger} , \tag{137}$$

where $\mathbf{A}^{\dagger}$ denotes the pseudoinverse of $\mathbf{A}$;

3. If $\hat{\mathbf{V}}_{22}$ is singular, chose a truncation parameter $s \leq \min(p, \operatorname{rank}(\Xi))$, such as $\hat{\mathbf{V}}_{22}$ is full rank and $\hat{\sigma}_s \neq \hat{\sigma}_{s+1}$, and then calculate the *truncated* total least squares solution, i.e.,

$$\hat{\boldsymbol{\beta}}_{ttls}^{*} = -\hat{\mathbf{V}}_{s\_12}\left(\hat{\mathbf{V}}_{s\_22}\right)^{\dagger} , \tag{138}$$

with the reduced (or truncated) matrix

$$\hat{\mathbf{V}}_{s} = \begin{bmatrix} \hat{\mathbf{V}}_{s\_11}^{(p\times s)} & \hat{\mathbf{V}}_{s\_12}^{(p\times q)} \\ \hat{\mathbf{V}}_{s\_21}^{(q\times p)} & \hat{\mathbf{V}}_{s\_22}^{(d\times q)} \end{bmatrix} , \tag{139}$$

with $q = p - s + d$. In this way, by truncated total least squares it is possible to solve numerically rank deficient problems, i.e., when the matrix $\boldsymbol{\Xi}$ has one or more small singular values, for instance when multicollinearity is present. By removing the small singular values of $\boldsymbol{\Xi}$, then a unique minimal norm solution $\hat{\boldsymbol{\beta}}^{*}_{ttls}$ can be found. The threshold s should be adaptively defined by some iterative process. In this work, it is adopted a similar approach for removing principal components, i.e., by normalizing singular values

$$\bar{\sigma}_i = \frac{\hat{\sigma}_i}{\max\left(\hat{\sigma}_i\right)} \ , \tag{140}$$

and then removing the proportionally smaller ones. For example, by setting a target value $\bar{\sigma}^* > 10\%$, then all normalized singular values $(\bar{\sigma}_i < \bar{\sigma}^*)$ will be removed from the solution. Another possibility is to use a procedure similar to the global cross-validation, GCV, and to remove the singular values one by one. The selected solution is the one with minimum GCV. In this case, since we have no parameter k, the GCV score is proportional to the norm of the error, i.e., $\left\| \mathbf{y} - \mathbf{X}\hat{\boldsymbol{\beta}}_{ttls} \right\|$.

Finally, in our context, the total least squares approach can be useful to improve the accuracy of ensemble of metamodels by least squares in two ways. First, although the original variables $\mathbf{x}$ for computer models are supposed to be deterministic and error free, the least squares solution for ensembles is defined based on modified variables, i.e., the approximate models $\hat{y}_i\left(\mathbf{x}\right)$, that are stochastic realizations and, as consequence, subjected to random errors which add noise to the matrix $\hat{\mathbf{Y}}$ in Equation (20). Second, when the multicollinearity is a concern, this problem can be assessed by regularization procedure like the truncated total least squares or any available variant, as discussed in Fierro and Bunch (1994) and Fierro and Bunch (1997).

G Manuscripts Submitted

We include in this appendix the manuscript versions of the papers submitted for publication.

The first paper is already published and the original file can be found at the journal's website. In this appendix we include for reference the the accepted manuscript version.

Ferreira W.G., Serpa A.L. (2016) *Ensemble of metamodels: the augmented least squares approach.* Structural and Multidisciplinary Optimization, 53(5), 1019-1046, 2016. DOI: 10.1007/s00158-015-1366-1

The second paper is still under review by the journal editors and reviewers. In this appendix we include for reference the the last submitted manuscript version.

Ferreira and Serpa (SMO-15-0339) *Ensemble of metamodels: Extensions of the least squares approach to efficient global optimization.* Structural and Multidisciplinary Optimization (submitted/under review - ID SMO-15-0339.R1)

Structural and Multidisciplinary Optimization manuscript No.
(will be inserted by the editor)

Ensemble of Metamodels: The Augmented Least Squares Approach

Wallace G. Ferreira · Alberto L. Serpa

Received: date / Accepted: date

Abstract In this work we present an approach to create ensemble of metamodels (or weighted averaged surrogates) based on least squares (LS) approximation. The LS approach is appealing since it is possible to estimate the ensemble weights without using any explicit error metrics as in most of the existent ensemble methods. As an additional feature, the LS based ensemble of metamodels has a prediction variance function that enables the extension to the efficient global optimization. The proposed LS approach is a variation of the standard LS regression by augmenting the matrices in such a way that minimizes the effects of multicollinearity inherent to calculation of the ensemble weights. We tested and compared the augmented LS approach with different LS variants and also with existent ensemble methods, by means of analytical and real-world functions from two to forty-four variables. The augmented least squares approach performed with good accuracy and stability for prediction purposes, in the same level of other ensemble methods and has computational cost comparable to the faster ones.

Keywords Ensemble of metamodels · Weighted average surrogates · Least squares approximation

W. G. Ferreira
Ford Motor Company Brazil
CAE & Optimization Engineering
Av. Taboão, 899
09655-900, S. B. Campo, SP, Brazil
Tel.: +55-11-41744207
E-mail: wferrei7@ford.com - wgferreira@yahoo.com

A. L. Serpa
University of Campinas - UNICAMP
School of Mechanical Engineering - FEM
Department of Computational Mechanics - DMC
13083-970, Campinas, SP, Brazil
Tel.: +55-19-35213387
E-mail: serpa@fem.unicamp.br

1 Introduction

In the last three decades, the use of metamodeling methods (also known as *surrogate modeling* or *response surface methodology*) to replace expensive computer simulation models such as FE (Finite Elements) or CFD (Computational Fluid Dynamics) found in automotive, aerospace and oil-gas industry, for example, has become a common place in both research and practice in engineering design, analysis and optimization. Extensive reviews in this area can be found in: Queipo et al (2005), Simpson et al (2008), Forrester and Keane (2009), Viana et al (2010) and Ramu and Prabhu (2013). A collection of engineering research and applications has been recently published in Koziel and Leifesson (2013).

The effectiveness of *selecting* and/or *combining* different metamodels in the optimization process has been investigated and discussed in the last years. See, for instance Zerpa et al (2005), Goel et al (2007), Sanchez et al (2008), Viana et al (2009), Acar and Rais-Rohani (2009), Acar (2010) and Viana (2011). In most of these studies, it is suggested that *ensemble of metamodels*, or *weighted averaged surrogate* (WAS) models are able to provide better accuracy than individual metamodels working alone, and can improve the overall robustness of the metamodel based optimization process. Besides that, in Viana et al (2009) and Viana (2011) it is discussed that the potential gains of using multiple surrogate models are not guaranteed and should be limited.

In fact, as pointed out in Viana (2011), “even after years of intensive research, surrogate modeling still involves a struggle to achieve maximum accuracy within limited resources”. In addition, as discussed by Yang et al (2013), among the challenges in modeling and optimization in the next years, (i) the best way to construct good surrogate models and (ii) the choice of modeling and optimization algorithms for a given problem are still open questions for research.

In this research, we performed a series of numerical experiments by applying the concept of least squares (LS) regression in order to find the optimal weights in ensemble of metamodels. In most of the currently available methods, for instance Goel et al (2007), Viana et al (2009) and Acar and Rais-Rohani (2009), it is necessary to use specific error metrics like PRESS (prediction sum of squares) to drive the process of finding the best weights in the ensemble of metamodels. Therefore, the LS approach is appealing in terms of lower computational cost and simple formulation. In addition, LS methods are well known and established in statistics field and have a series of variants developed to handle, for example, multicollinearity, an inherent drawback that limits the accuracy of ensemble of metamodels.

Specifically for handling multicollinearity, we propose in this work the *augmented least squares ensemble of metamodels*, a variation of the standard least squares regression, by augmenting the matrix system in such a way that minimizes the effects of linear dependency among the models, inherent to calculation of the ensemble weights, specially when the least squares approach is applied.

In a second front of application, LS ensemble methods can be used within the context of efficient global optimization. The ensemble of metamodels constructed based on LS approach inherits the variance estimator, which can be used in the definition of the *expected improvement function*. In summary, efficient global optimization (EGO) (ref. Jones et al (1998)) is an iterative approach that, at each optimization cycle defines one *infill* point that maximizes the expected improvement, with respect to the minimum of the objective function. Recently, Viana et al (2013) presented the MSEGO algorithm (multiple surrogates EGO), that extends the concept of EGO by using multiple metamodels to generate several infill points in parallel.

The aim of our research is to present the findings regarding the augmented least squares approach for ensemble of metamodels in both *prediction* and *optimization* purposes. In order to fulfill this objective, we divided research in two manuscripts. In the present one we will focus on the development of the LS ensemble of metamodels algorithm and the numerical experiments performed to verify the accuracy of the proposed method for *prediction*. The *optimization* objectives will be covered in a companion work, Ferreira and Serpa (2015b), in which we will present the extensions and results of the proposed approach in the context of EGO algorithms and applications.

The remainder of the present manuscript will be divided as follows. In Section 2, the metamodeling process and the metrics used for model validation are outlined. The previous research in the field of ensemble of metamodels is presented in Section 3. The concept of ensemble of metamodels by least squares approximation and the proposed augmented least squares approach are presented and detailed in Section 4. In Section 5 we present and discuss the results of the numerical experiments performed to validate and compare the augmented least squares approach with other ensemble methods. Finally, the concluding remarks are presented in Section 6.

2 The Metamodeling Process

Most physical phenomena can be described by using a mathematical model, such as $y = f(\mathbf{x})$, where $\mathbf{x}$ represent an input vector of variables defined in $\Re^n$, and y is an output scalar variable that represents the response $f(\mathbf{x})$ being modeled. Since $f(\mathbf{x})$ is costly and/or time consuming to evaluate, as in computer simulation models (e.g., FE and CFD) for real world applications, the idea is to find a proper approximation $\hat{y}(\mathbf{x}) \approx y(\mathbf{x})$, also known as metamodel, surrogate or response surface, that is accurate, cheap and fast to evaluate.

The iterative process of constructing approximate models (i.e., metamodeling) should be described briefly in five steps, as follows:

(i) *Definition of Design Space*: select the n_v design variables $\mathbf{x} = [x_1 \ \cdots \ x_{n_v}]^T$ and define the bounds of the design space, i.e., $\mathbf{x} \in \chi$. This step can be driven by prior experience and knowledge regarding the problem, or by a previous sensitivity analysis to prioritize the most significant n_v variables and respective bounds;

(ii) *Experimental Design*: perform a design of experiments (DOE), e.g., Factorial Design, Uniform Design, Latin Hypercube, etc., and generate a set of N sampling points (or "training" points) to evaluate the true function $y(\mathbf{x})$;

(iii) *Models Evaluation*: run the true models and evaluate $y(\mathbf{x})$ at the training points defined in step (ii);

(iv) *Metamodels Creation*: create the metamodel $\hat{y}(\mathbf{x})$, by using any available procedure, e.g.: PRS, Polynomial Response Surface; RBF, Radial Basis Functions; KRG, Kriging; NN, Neural Networks; SVR, Support Vector Regression, etc.;

(v) *Metamodel Validation*: define a suited error metric and test/validate the accuracy of $\hat{y}(\mathbf{x})$. If the accuracy is not acceptable, iterate by adding more sampling points, step (ii); or by improving the approximation method, step (iv).

In this way, by means of validated metamodels $\hat{y}(\mathbf{x})$, many engineering activities can be performed in a faster way, such as: preliminary studies and graphical visualization; prediction and optimization; design sensitivity analysis; probabilistic, robust and reliability based design.

The metamodeling process for design and optimization has been developed and matured during the last three decades. All the five steps (i-v) outlined above are reasonably well established in engineering design, and the details can be found in the books of Fang et al (2006) and Forrester et al (2008).

2.1 Metrics for Metamodel Validation

Regarding the step (v) in the metamodeling process outlined previously (i.e., validation), several metrics can be defined to evaluate the prediction error $e(\mathbf{x}) = y(\mathbf{x}) - \hat{y}(\mathbf{x})$, for example: maximum absolute, average, correlation coefficient, etc. One common error measure adopted is the *root mean squared error*, defined by

$$RMSE = \sqrt{\frac{1}{N_{test}} \sum_{i=1}^{N_{test}} e_i^2(\mathbf{x})} \ , \tag{1}$$

where N_{test} is the number of test points used to evaluate the prediction error.

Another common strategy, known as *cross-validation*, is defined as follows. For $i = 1, \cdots, N$, let $\hat{y}_{-i}$ denote the metamodel constructed based on excluding the i-th sampling point $(\mathbf{x}_i, y_i)$ from the original set (*leave-one-out*). Therefore, the cross validation score or PRESS (PREdiction Sum of Squares) is given by

$$CV_N = \frac{1}{N} \sum_{i=1}^{N} \{y(\mathbf{x}_i) - \hat{y}_{-i}(\mathbf{x}_i)\}^2 . \tag{2}$$

In order to reduce computational cost in the cross-validation process, the set of sampling points N can be divided into k groups of the same size, and the procedure is calculated by removing all points in each k-th group, instead of one single point by time. Therefore, there is a trade-off between accuracy and computational cost to estimate PRESS via this k-fold cross-validation.

3 Ensemble of Metamodels Background

3.1 Origins and Definitions

Concepts like *ensemble of predictors*, *mixture of experts*, *committee of networks*, etc., are well known in machine learning literature. See for instance Wolpert (1992), Hashem (1993), Perrone and Cooper (1993), Bishop (1995) and Breiman (1996). The idea is that, by the combination of different predictors, it is possible to improve the final model accuracy.

In this sense, an ensemble of models is defined as linearly weighted summation[1]

$$\hat{y}_{ens}(w_i \ , \mathbf{x}) = w_0 + \sum_{i=1}^{M} w_i \hat{y}_i(\mathbf{x}) \ , \tag{3}$$

where $\hat{y}_i(\mathbf{x})$ are the M available distinct models, w_i are the weights associated in the linear combination and w_0 is an intercept term. If $w_0 = 0$ and $w_i = \dfrac{1}{M}$, then it is defined the *simple average* ensemble (SA).

It is acknowledged to Perrone and Cooper (1993) and Hashem (1993) the first developments on finding *optimal weights* w_i in the ensemble. They devised independently an approach that has been presented and discussed in detail in the book by Bishop (1995). In summary, by the minimization of the mean squared error (MSE) with respect to w_i and with $\sum_i w_i = 1$ as constraint, it follows that

$$w_i = \frac{\sum_{i=1}^{M} \left(\mathbf{C}^{-1}\right)_{ij}}{\sum_{k=1}^{M} \sum_{j=1}^{M} \left(\mathbf{C}^{-1}\right)_{kj}} \ , \tag{4}$$

where the error correlation matrix $\mathbf{C}$ is estimated based on a validation set of N_v sampling points as follows

$$C_{ij} \approx \frac{1}{N_v} \sum_{n=1}^{N_v} \left[(y_n - \hat{y}_i(\mathbf{x}_n)) (y_n - \hat{y}_j(\mathbf{x}_n)) \right] . \tag{5}$$

It can be demonstrated that (ref. to Perrone and Cooper (1993) or Bishop (1995) for details)

$$MSE(\hat{y}_{ens}) \leq \frac{1}{M} \sum_{i=1}^{M} MSE(\hat{y}_i) \, , \tag{6}$$

which means that: (i) the higher the number of distinct models, the lower is expected the MSE for the weighted average ensemble (with a factor of $\frac{1}{M}$); and (ii) $\hat{y}_{ens}(\mathbf{x})$ provides the best estimate of $y(\mathbf{x})$ in the mean square sense, if the errors $e_i = y_n - \hat{y}_i(\mathbf{x}_n)$ have zero mean and are uncorrelated. In practical applications, these theoretical levels of MSE reduction are difficult to achieve because the models $\hat{y}_i(\mathbf{x})$ can be highly correlated. In

[1] Most of the publications is focused on linear ensembles, but it can be observed a growth of interest on *nonlinear ensemble methods*, in which any type of approximation should be used to combine the models, e.g., neural networks, support vector regression, etc. See, for instance Yu et al (2005), Lai et al (2006) and Meng and Wu (2012).

spite of that, we can find in the literature results advocating 10% or higher for reduction in MSE by using weighted ensemble of predictors.

Motivated by the potential benefits, ensemble methods for prediction is still an active area of research that is getting maturity, as can be seen in the recent books dedicated to this topic: Seni and Elder (2010), Zhou (2012) and Zhang and Ma (2012).

Although ensemble methods are well known in machine learning area, specially in predictive statistics and financial forecasting, this approach is relatively new in the engineering field. This is what we will present and discuss with more detail in the next section.

3.2 Ensemble Methods in Engineering Design

The first application of ensemble methods in engineering design and optimization, inspired by the work in machine learning, is acknowledged to Zerpa et al (2005). They proposed the estimation of the weights w_i in the linear ensemble, Eq. (3), as

$$w_i = \frac{\dfrac{1}{V_i}}{\displaystyle\sum_{j=1}^{M} \frac{1}{V_j}} \ , \tag{7}$$

where V_i is the prediction variance estimation $V\left(\hat{y}\left(\mathbf{x}\right)\right)$ for the i-th metamodel.

Goel et al (2007) proposed heuristic schemes for estimating the weights in a weighted averaged surrogate model (WAS), for instance: PWS (PRESS weighted surrogate), given by

$$w_i^{PWS} = \frac{w_i^*}{\displaystyle\sum_{j=1}^{M} w_j^*} \ , \tag{8}$$

where w_i^* is defined as

$$w_i^* = \left(E_i + \alpha E_{avg}\right)^{\beta} \quad \text{with} \quad E_{avg} = \frac{1}{M}\sum_{j=1}^{M} E_j \ , \tag{9}$$

in which E_i is the global data-based error measure for the i-th surrogate model, in this case PRESS. The parameters $(\alpha < 1$ and $\beta < 0)$ control the importance of averaging and the importance of an individual surrogate, respectively and it was suggested $\alpha = 0.05$ and $\beta = -1$.

Acar and Rais-Rohani (2009) estimated the weight factors by solving a direct optimization problem of the form:

$$\begin{cases} \min\limits_{w_i} \ E_{rr}\left(\hat{y}_{ens}\ , y\right) \\ \text{subjected to (s.t.)} \ \displaystyle\sum_{j=1}^{M} w_j = 1 \ , \end{cases} \tag{10}$$

and $E_{rr}\left(\hat{y}_{ens}\ , y\right)$ is any selected error metric. In this case it was used RMSE and PRESS to drive the optimization.

Viana et al (2009) defined the optimal weighted surrogate (OWS) by:

$$\min_{w_i} \ \ \text{MSE}\left(\hat{y}_{ens}\right) = \mathbf{w}^T\mathbf{C}\mathbf{w}, \quad \text{s.t.} \ \sum_{j=1}^{M} w_j = 1, \tag{11}$$

with the matrix $C_{ij} = \dfrac{1}{N}\tilde{\mathbf{e}}_i^T\tilde{\mathbf{e}}_j$, where $\tilde{\mathbf{e}}$ is the vector of cross-validation errors (PRESS) for the i-th and j-th surrogates. The matrix $\mathbf{C}$ is similar to the one in Eq. (4), but in Viana et al (2009) it is modified by using cross-validation procedure, Eq. (2) , with the whole set of N sampling points.

In all the applications mentioned up to here the weights w_i are considered constant in the design domain (global weights). There is no restriction on defining the weights as dependent on the location of the sampling points (local weights), i.e., $w_i\left(\mathbf{x}\right)$. Even though, the results published with local weights did not show remarkable improvements, when compared with the constant definition of weights in the design domain. See for example Sanchez et al (2008) and Acar (2010) for more details.

In general, most of the previous research suggests that ensemble of metamodels, or weighted averaged surrogate models (WAS), are able to provide better accuracy than individual metamodels working alone. In other words, it is advocated that weighted averaging schemes should improve the robustness of the predictions and the optimization results, by reducing the impact of poorly fitted surrogates in the ensemble.

On the other hand, as discussed in Viana et al (2009), the computational cost to calculate PRESS can become prohibitive and, in addition, the gains in terms of reduction of RMSE diminishes substantially as the number of variables increases, even with a large number of sampling points. According to this research, on the contrary as stated in Eq. (6), none of the ensemble methods tested was able to reduce the RMSE more than 10% as compared to the best model (i.e., the most accurate in terms of PRESS). In fact, in some cases they found that the ensemble model can be less accurate than the best model. Therefore, for the problems tested, it was not verified enough evidence that the combination of models is always better than selecting the best model, at least when PRESS is used as error metric to rank models and drive the estimation of the weights in the ensemble.

Finally, since there are still some controversy and open questions, metamodeling methods for prediction and optimization are included in the list of challenges for research in the next years, as discussed by Yang et al (2013).

4 Ensemble of Metamodels by Least Squares

4.1 Basic Formulation

Here it will be followed an alternative approach for calculating weights in an ensemble model, by minimizing the approximation error, without explicitly calculating any error measure (e.g., PRESS), as discussed in the last section.

Let us recall that a general linear regression model (ref. Montgomery et al (2006)) can be written as

$$y = \beta_0 + \sum_{i=1}^{n_{reg}} \beta_i z_i(\mathbf{x}) + \varepsilon \ , \tag{12}$$

where $z_i(\mathbf{x})$ represents any function of the original *regressors* (or design variables) x_j, and ε is the error on the approximation.

Then, by replacing $z_i(\mathbf{x}) = \hat{y}_i(\mathbf{x})$ and $\beta_i = w_i$, the linear ensemble in Eq. (3) can be rewritten as a linear regression problem, i.e.,

$$\mathbf{y} = \hat{\mathbf{Y}}\mathbf{w} + \boldsymbol{\varepsilon}, \tag{13}$$

where $\mathbf{y} = [y_1 \ \cdots \ y_N]^T$, $\hat{\mathbf{Y}} = [\hat{y}_1(\mathbf{x}_i) \ \cdots \ \hat{y}_M(\mathbf{x}_i)]$ and $\mathbf{w} = [w_1 \ \cdots \ w_M]^T$, for N sampling points and M metamodels.

Therefore, *standard least squares estimator* for $\mathbf{w}$ in Eq. (13) is given by

$$\hat{\mathbf{w}} = (\mathbf{Y}^T\hat{\mathbf{Y}})^{-1}\hat{\mathbf{Y}}^T\mathbf{y}, \tag{14}$$

and, according to the *Gauss-Markov* theorem, if the errors $\boldsymbol{\varepsilon} = [\varepsilon_1 \ \cdots \ \varepsilon_N]^T$ are normally and independently distributed, with zero mean and finite variance, i.e., $\varepsilon_i \sim \text{NID}(0, \sigma^2)$, then $\hat{\mathbf{w}}$ is the referred as the *best linear unbiased estimator* (BLUE) of $\mathbf{w}$. That is, $\hat{\mathbf{w}}$ provides the minimum prediction error, in the least squares sense, and has minimum variance among all unbiased linear estimators. For proofs and details see Montgomery et al (2006) or Björk (1996).

The solution of the $N \times M$ set of linear equations in Eq. (14) is well known and can be performed efficiently by standard numerical algorithms (ref. Björk (1996)). In this way, $\hat{\mathbf{w}}$ minimizes the errors ($e_i = y_i - \hat{y}^i_{ens}$), in the least squares sense, without explicitly calculating any costly error measure, like PRESS for instance.

In addition to the simple formulation and computational efficiency, another interesting property of ensemble methods with optimal weights estimated by Eq. (14) is that they inherit the *least squares variance estimate* $V[\hat{y}_{ens}(\mathbf{x})] \equiv \hat{s}^2(\mathbf{x})$, for the prediction at each point $\mathbf{x}$, that can be written as

$$\hat{s}^2(\mathbf{x}) = \hat{\sigma}^2 [\hat{\mathbf{y}}(\mathbf{x})]^T \left(\hat{\mathbf{Y}}^T\hat{\mathbf{Y}}\right)^{-1} \hat{\mathbf{y}}(\mathbf{x}) \ , \tag{15}$$

with $\hat{\mathbf{y}}(\mathbf{x}) = [\hat{y}_1(\mathbf{x}) \ \hat{y}_2(\mathbf{x}) \ \cdots \ \hat{y}_M(\mathbf{x})]^T$, and

$$\hat{\sigma}^2 = \frac{\hat{\mathbf{y}}^T_{ens}\hat{\mathbf{y}}_{ens} - \hat{\mathbf{w}}^T\hat{\mathbf{Y}}\hat{\mathbf{y}}_{ens}}{N - n_v} \tag{16}$$

where $\hat{\mathbf{y}}_{ens} = [\hat{y}_{ens}(\mathbf{x}_1) \ \hat{y}_{ens}(\mathbf{x}_2) \ \cdots \ \hat{y}_{ens}(\mathbf{x}_N)]^T$.

As we mentioned in Section 1, by means of the variance estimate, Eq. (15), it is possible to derive an expected improvement function, which is the main ingredient for the application of efficient global optimization algorithms. This branch of the research will be covered in the companion work Ferreira and Serpa (2015b).

It is worth noting that the idea of using least squares regression to find the optimal ensemble weights in fact is not new. As remarked by Hashem (1993), if it is removed the constraint to derive the Eq. (4), the solution for the optimal weights is equivalent to the least squares regression. In spite of that, in the work by Hashem (1993) it was only applied and tested the method based on Eq. (4). In addition, the authors remarked that, due to the intrinsic similarity in the metamodels $\hat{y}_i$, then "potential problem" of multicollinearity can take place and it can jeopardize the final accuracy of the ensemble predictor. Despite this observation, no further suggestion or investigation on methods to solve (or at least reduce) the multicollinearity issue have been addressed in the work by Hashem (1993) or even in the related research by Perrone and Cooper (1993) or Bishop (1995).

By means of some preliminary numerical tests with analytical functions (up to ten variables, $n_v = 10$), we found that the LS ensemble method is comparable in terms of accuracy (same level of average RMSE) and it performs much faster (one order of magnitude or higher) than the PRESS based ensemble methods available, as presented in Section 3.2). On the other hand, we also observed that LS ensemble can become unstable (i.e., high variation on RMSE with different data), specially as the number of variables increases. By using a *stepwise* selection procedure in the LS solution, we found around 30% reduction in the standard deviation of RMSE for the tests performed. Therefore, we got some evidence that, by combating multicollinearity, it is possible to improve the accuracy of the final LS ensemble of metamodels. See Appendix B for details.

These preliminary results motivated us to a deeper investigation regarding other available variants of least squares methods to handle multicollinearity (e.g., ridge regression, principal components, etc.), in order to understand if it is possible to further improve the accuracy and stability of the LS approach. In addition, we aim to understand how the LS ensemble compares to the available ensemble methods developed up to now. As far as we could investigate, there was not found any similar work in the literature in this sense and we aim to contribute in this front.

4.2 The Augmented Least Squares Approach

The issue of multicollinearity in least squares regression is well known in statistics and related areas. See for instance Montgomery et al (2006), Chap. 11, and the list of references therein for a broad perspective on this subject. Among the several sources of multicollinearity, the primary ones are: (i) the data collection method (size and distribution of sampling points) an (ii) models with redundant variables (over-defined). During the last decades, several methods were devised for dealing with multicollinearity in least squares problems. In general, the techniques involve: a) *gathering additional data* and b) *model respecification*, in order to reduce the prediction errors induced by multicollinearity.In Sec. 4.3 the most traditional methods are summarized. In addition, see Appendix C for an introductory background on this subject.

In the present section we suggest an approach based on the idea of gathering additional data to reduce multicollinearity. In Fig. 1 we have the true response $y(x)$ (solid line), and four different metamodels $\hat{y}_i(x)$ (dashed lines). It can be observed that the prediction at the $N = 8$ sampling points (circles), used to generate the approximations, is similar for all metamodels, specially in case of interpolating ones, in which the prediction is the same by definition. On the other hand, the prediction is much more different in the additional, $N_{add} = 7$, out-of-sample points (stars), which were not used to generate the metamodels.

Let us consider two different cases. In the first one, the N points are used to generate the metamodels $\hat{y}_i(x)$ and then the matrix $\hat{\mathbf{Y}} = [\hat{y}_1(x_i) \ \cdots \ \hat{y}_M(x_i)]$ is assembled. In the second case, the metamodels $\hat{y}_i(x)$ are evaluated at the N_{add} points, and an *augmented* matrix $\hat{\mathbf{Y}}_{aug}$ is assembled as follows

$$\hat{\mathbf{Y}}_{aug} = \left[\begin{array}{ccc} \hat{y}_1(\mathbf{x}_1) & \cdots & \hat{y}_M(\mathbf{x}_1) \\ \vdots & \ddots & \vdots \\ \hat{y}_1(\mathbf{x}_N) & \cdots & \hat{y}_M(\mathbf{x}_N) \\ \hline \hat{y}_1(\mathbf{x}_{N+1}) & \cdots & \hat{y}_M(\mathbf{x}_{N+1}) \\ \vdots & \ddots & \vdots \\ \hat{y}_1(\mathbf{x}_{N+N_{add}}) & \cdots & \hat{y}_M(\mathbf{x}_{N+N_{add}}) \end{array}\right].$$

Since the predictions in the augmenting N_{add} points are expected to differ among the metamodels (as in the

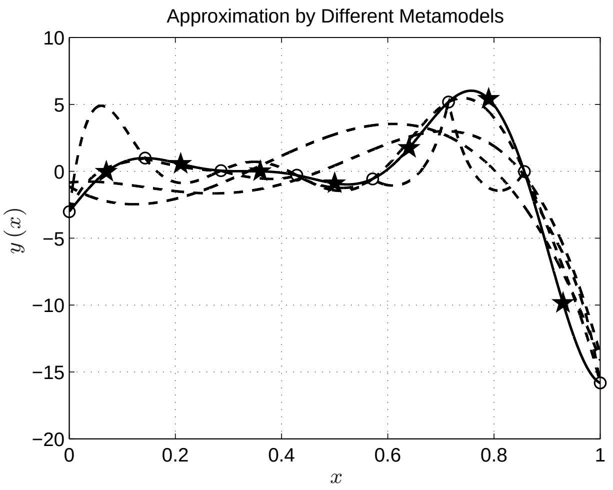


Fig. 1 Combining multiple metamodels by different methods. Continuous line: true response $y(x) = -(6x-2)^2 \sin(12x-4)$; Dashed lines: metamodels by PRS, KRG, NN and SVR; Circles: sampling points used to create the metamodels; Stars: midpoints, out-of-sample data not used in the approximations.

star points in Fig. 1), it is reasonable to assume that any collinearity among the columns of $\hat{\mathbf{Y}}$ has a good chance to be reduced in the augmented case $\hat{\mathbf{Y}}_{aug}$, if we have enough N_{add} points to take advantage of the diversity in the metamodel predictions in points out-of-sample.

If this conjecture is valid, the ensemble weights $\mathbf{w}$ can be estimated for the augmented system as

$$\hat{\mathbf{w}}_{aug} = \left(\hat{\mathbf{Y}}_{aug}^T \hat{\mathbf{Y}}_{aug}\right)^{-1} \hat{\mathbf{Y}}_{aug}^T \mathbf{y}_{aug}, \tag{17}$$

with $\mathbf{y}_{aug} = \left[y_1 \ \cdots \ y_N \mid y_{N+1} \ \cdots \ y_{N+N_{aug}}\right]^T$. Since the linear dependency in $\hat{\mathbf{Y}}_{aug}$ is expected to be lower than in $\hat{\mathbf{Y}}$, then the accuracy and stability on estimating $\mathbf{w}$ by means of $\hat{\mathbf{w}}_{aug}$ is expected to be improved as well.

In many practical situations of metamodeling in engineering design, gathering additional points should be time consuming, costly or even impossible. In these cases, the whole data set of N sampling points available should be split in two parts, for example $N = N_{tr} + N_{add}$. In the sequence, the metamodels are created based on the *training* data set N_{tr} and the weights are then calculated based on the augmented system, Eq. (17), with all the N points available. Therefore, in this case it will arise a trade-off regarding the loss of accuracy in the metamodels approximated with less points (i.e., $N_{tr} < N$) and the possible gains of stability with the augmented approach. In this sense, the optimal number of augmenting points, or the ideal rate $\eta_{aug} = \dfrac{N_{add}}{N_{tr} + N_{add}}$, to balance accuracy and stability in the least squares ensemble needs be investigated.

In summary, the augmented least squares approach can be outlined in steps as follows:

i. Collect $N = N_{tr} + N_{add}$ random sampling points and evaluate the true response y for all points collected;
ii. Generate the approximation models $\hat{y}_i$ by using only the N_{tr} sampling points;
iii. Evaluate the metamodels $\hat{y}_i$ at all $N = N_{tr} + N_{add}$ points and mount the augmented matrix $\hat{\mathbf{Y}}_{aug}$ and the output vector $\hat{\mathbf{y}}_{aug}$;
iv. Solve the system for the weights $\hat{\mathbf{w}}_{aug}$ with Eq. (17).

As we stated before, part of our objective is the comparison of the proposed augmented LS approach among other LS variants developed to handle multicollinearity. In this sense, we will present in the next section LS variants based on model re-specification, that were developed during the decades in order to reduce the prediction errors, specially the ones induced by multicollinearity.

In addition, we will compare our proposed method against the previous ones, summarized in Sec. 3.2. Although multicollinearity is not clearly discussed in deep on these previous works, it can be observed that in some sense most of them also try to combat the multicollinearity effects in the ensemble, for instance by smoothing the variance of the predictors, as in Eq. (7), or by controlling the *size* of the weights by the constraint $\sum_{j=1}^{M} w_j = 1$, as in Eq. (10) and Eq. (11). On the other hand, these kind of control is not clear in PWS, given by Eq. (8).

4.3 Least Squares Regression Variants

For brevity we will not present here the detailed formulation for all the LS variants. Most of them are covered in Montgomery et al (2006) and details regarding numerical implementation are discussed by Björk (1996). Further information, assumptions and proofs can be found in the references listed in each topic. See Appendix C a brief background on the the main motivations that drive these methods.

Stepwise Selection: algorithms developed to add and/or delete variables in a regression model to control the accuracy and to reduce sources of error like multicollinearity. In Miller (2002) it is presented an extensive discussion on these methods. We followed the implementation presented in Fang et al (2006) for the variants: F-statistic by Efroymson (1960); Akaike Information Criterion (AIC) by Akaike (1974); Bayesian Information Criterion (BIC) by Schwarz (1978); φ-Criterion by Hannan and Quinn (1979) and Shibata (1984); and the Residual Information Criterion (RIC) by Foster and George (1994);

Ridge Regression: originally published in the two companion papers Hoerl and Kennard (1970a) and Hoerl and Kennard (1970b) to handle nonorthogonal data (i.e., multicollinear). Several algorithms have been proposed in the literature for finding the best ridge estimator, as the *generalized cross-validation* (GCV) procedure by Golub et al (1979), that is one of the most accepted for automation;

Principal Components Regression: a deep and extensive discussion on principal components analysis can be found in Jolliffe (2002). In summary, the original variables x_i are transformed to the principal components (PC) space z_i and then the less significant PC can be identified and removed to reduce the multicollinearity in the system. As in ridge regression, several algorithms have been proposed as well for removing the less significant PC. Golub et al (1979) proposed a variant of GCV suited to principal components selection and it is applied here;

Constrained and Penalized Least Squares: in this cathegory, constraints should added to the ordinary least squares problem, for example the *sum to unit* and/or the *positivity* of the coefficients, i.e., $\sum\limits_{i=1}^{p} \beta_i = 1$ and $\beta_i \geq 0$ and the problem now is known as *constrained least squares* regression. In order to unify different constrained approaches, a general *penalized least squares* can be defined (ref. Fan and Li (2001) and Fang et al (2006)). The LASSO (least absolute shrinkage and selection) algorithm, by Tibshirani (1996) and the SCAD (smoothly clipped absolute deviation) by Fan and Li (2001) are in this category. We followed the implementation presented in Fang et al (2006);

Generalized and Weighted Least Squares: the ordinary least squares equation is modified by adding a weighing matrix, in order to control the variance of the regression coefficients. The most common procedure is the *iteratively re-weighted least squares*, IRLS. We followed the procedure presented in Montgomery et al (2006). See Weisberg (1985) and Amemiya (1985) for different weighing schemes.

Robust Regression: these variants are adopted when the data are not normally distributed or there are possible *outliers*. The literature regarding robust regression is vast and a lot of classes of robust procedures have been proposed (ref. Rousseeuw and Leroy (2003) and Huber and Rochetti (2009)). We adopted the class *M-Estimators* presented in Montgomery et al (2006);

Total Least Squares: if both input and output matrices are contaminated by noise, the ordinary least squares solution can not be accurate (ref. van Huffel and Vandewalle (1991) and Markovsky and van Huffel (2007)). When *total least squares* method is applied, both errors in inputs and outputs are minimized in the Frobenius norm sense. The basic algorithm to solve total least squares is detailed in Fierro and Bunch (1997).

4.4 Effect of the Intercept Term

The majority of ensemble methods based on the definition in Eq. (3) does not take into account the intercept term for generating the linear ensemble of models, i.e., it is assumed $w_0 = 0$.

In Hashem (1993) it was remarked the effect of the intercept in the accuracy of the final ensemble of neural networks. They identified that for well-trained network models, the optimal sum of weights tends to one and the constant term tends to zero. However, for poorly trained networks the sum of optimal weights is far from one and the constant term can be significantly different from zero.

In linear regression analysis this effect of intercept term is also known. As discussed by Montgomery et al (2006), the effect of intercept can be significant, specially when data lie in a region far from the origin of the design space.

Therefore, there is no reason *a priori* to neglect the intercept term in the ensemble models and it can be a good practice to check whether the intercept term improves the accuracy of approximation or not.

5 Numerical Experiments

In this section we present the numerical experiments performed to validate and compare the performance proposed augmented LS approach. In summary, the specific objectives are:

- to investigate the ideal number of augmenting points, or the rate η_{aug}, to create the LS ensemble;
- to verify the effect of the number of metamodels in the set to generate the LS ensemble;
- to compare the augmented LS approach with the existent LS variants;
- to verify the effect of the intercept term in the accuracy of the final LS ensemble model;
- to compare the augmented LS approach with the existent ensemble methods, in terms of accuracy and computational cost.

5.1 Computer Implementation

We used as platform for implementation and tests the SURROGATES Toolbox v2.0 (ref. Viana (2009)), which is a Matlab[2] based toolbox that aggregates and extends several open-source tools previously developed in the literature for design and analysis of computer experiments, i.e., metamodeling and optimization (including EGO variants at v3.0[3]). The SURROGATES Toolbox uses the following collection of third party software: SVM by Gunn (1997), DACE by Lophaven et al (2002), GPML by Rasmussen and Williams (2006), RBF by Jekabsons (2009), and SHEPPACK by Thacker et al (2010). The compilation in a single framework has been implemented and applied in previous research by Viana and co-workers, as for example Viana et al (2009) and Viana (2011).

We implemented Matlab routines for the LS variants, as described in Section 4, and incorporated them to the function `srgtsWAS`, i.e., the corresponding SURROGATES Toolbox function for creating WAS models (weighted averaged surrogates). For the ordinary LS solution and robust regression, we used the respective Matlab built-in functions `regress` and `robustfit`. We adapted the Matlab implementation by Fang et al (2006) for stepwise selection and penalized least squares. For all the other LS families and variants we developed our own algorithms, based on the references listed in Section 4.3.

[2] Matlab is a well known and widely used numerical programing platform and it is developed and distributed by The Mathworks Inc., see `www.mathworks.com`.

[3] Further details and recent updates of SURROGATES Toolbox refer to the website: `https://sites.google.com/site/srgtstoolbox/`.

Several LS variants were implemented and tested in preliminary studies. For brevity, we will present here only the most representative for each family. In Tab. 1, the respective acronyms and descriptions of the LS variants compared are presented.

Table 1 Least squares variants: acronyms and descriptions. As in LS-a, the suffix "-a" can be added to all LS variants to indicate the use in conjunction to the augmented approach.

Acronym	**Description**
LS	Ordinary Least Squares
LS-a	Augmented LS
S	Stepwise Selection, F-statistic
C	Constrained LS: $\sum_i w_i = 1$ and $w_i \geq 0$
R	Ridge Regression with GCV criterion
P	Principal Components with GCV criterion
W	Weighted Least Squares
T	Total Least Squares
Ro	Robust Regression with Huber function

The numerical implementation has been performed with Matlab v2009, on a computer Intel(R) Core(TM) i7-3610QM, CPU 2.30GHz, 8Gb RAM, 64bits, and operational system Windows 7.

5.2 Experimental Setup

Table 2 Basic specifications for the experiments. n_v: number of variables, N_{tr}: training points, N_{add}: augmenting points and $Ntest$: test points to calculate RMSE.

Test Problem	n_v	N_{tr}	N_{add}	N_{test}
Branin-Hoo	2	20	10	5×2000
Hartman	3	30	15	5×2000
Hartman	6	60	30	5×2000
Ext. Rosenbrock	9	110	10	5×2000
Dixon-Price	12	190	95	5×2000
Truck Durability	12	120	30	30
Truck Dynamics	12	120	30	30
Car NVH	30	100	30	30
Car Crash	44	200	50	50

The approach adopted here for the numerical experiments follows the methodology traditionally applied in the field. In particular, we combined the strategies used in Viana et al (2009) and Acar and Rais-Rohani (2009) to perform our tests.

The ensemble of metamodels were composed with four ($M = 4$) distinct models: PRS, KRG, RBNN and SVR, by considering the setup presented in Tab. 3. Refer to SURROGATES Toolbox manual, Viana (2009), for details on the equations and tuning parameters for each of these metamodeling methods.

We tested analytical benchmark functions and also responses from engineering applications in the range of number of variables from $n_v = 2$ to $n_v = 44$. The details regarding each test problem are presented in the Appendix A.

The quality of the approximation by metamodels is strongly dependent on the number and distribution of sampling points defined in the design space (i.e., design of experiments, DOE). As a common practice in comparative studies of metamodeling performance, in all the cases investigated, we repeated the each experiment with 100 different DOE, in order to reduce the influence of random data on the quality of fit by averaging the results. The detailed setup for the experiments with each test problem is presented in Tab. 2.

In case of the analytical benchmarks, the sampling points (N_{tr} and N_{add}) were created by using the Latin Hypercube Matlab function `lhsdesign`, optimized with `maxmin` criterion set to 1000 iterations. The RMSE, as defined in Eq. (1), is calculated based on the average RMSE of 5 test sets with $N_{test} = 2000$ points each. In case of the test set (N_{test}), `maxmin` is set to 10 iterations. For the engineering applications tested here, the full data set is fixed and it is not possible to generate 100 different DOE, as in the analytical benchmarks. In order to surpass this difficulty, in each of the 100 runs, the points for the data sets (N_{tr}, N_{add} and N_{test}) were chosen randomly from the original full set of sampling points. In all test problems, when applicable, the cross-validation procedure was performed with $k = 10$, to balance accuracy and computational cost for PRESS calculation.

The performance of different ensemble methods is compared based on the average (μ) and standard deviation (σ) of RMSE, over the 100 repetitions for the same test problem. We use in most cases *boxplots*[4] for easy visualization and comparison of the methods.

For convenience and easy reference, the competitor ensemble methods of Sec. 3, compared in this work, are summarized in Tab. 5.

[4] Boxplot is a common statistical graph used for visual comparison of the distribution of different variables in a same plane. The box is defined by lines at the lower quartile (25%), median (50%) and upper quartile (75%) of the data. Lines extending above and upper each box (*whiskers*) indicate the spread for the rest of the data out of the quartiles definition. If existent, outliers are represented by plus signs "+", above/below the whiskers. We used the Matlab function `boxplot` (with default parameters) to create the plots.

Table 3 Basic metamodels setup for creating the ensembles, when $M = 4$.

Model	**Modeling technique**	**Details**
PRS	Polynomial Response Surface	Full quadratic model
KRG	Kriging	Quadratic regression, exponential correlation, $\theta_0 = 10$ and $10^{-2} \leq \theta_i \leq 200$
RBNN	Radial Basis Neural Network	$Goal = (0.05\bar{y})^2$ and $Spread = 2/5$
SVR	Support Vector Regression	$C = 100max(\lvert\bar{y} + 3\sigma_y\rvert, \lvert\bar{y} - 3\sigma_y\rvert)$ and $\epsilon = \sigma_y/\sqrt{N}$

Obs.1: All other parameters not mentioned are kept with default values.
Obs.2: $\bar{y}$, σ_y and N are respectively: mean and standard deviation of y and number of sampling points.
Obs.3: No attempt has been made in order to fine tuning the surrogates shape parameters.

Table 4 Metamodels setup for creating additional the ensembles, when $M > 4$. Otherwise refer to Tab. 3.

ID/M	**Acronym**	**Details**
5	PRS	Linear model
6	RBNN	$Spread = 1/3$ and $MN = N/2$
7	KRG	Linear regression, Gaussian correlation
8	SVR	$C = 100max(\lvert\bar{y} + 3\sigma_y\rvert, \lvert\bar{y} - 3\sigma_y\rvert)$, $\epsilon = \sigma_y/\sqrt{N}$, $Loss = quadratic$ and $KernelOptions = 0.5$
9	RBNN	$Goal = (0.05\bar{y})^2$, $Spread = 1/2$ and $MN = N/3$
10	KRG	Linear regression, spherical correlation
11	KRG	Constant regression, cubic correlation
12	SVR	$C = \infty$, $\epsilon = 10^{-4}$ and $Loss = quadratic$
13	KRG	Quadratic regression, spline correlation
14	KRG	Linear regression, linear correlation
15	SVR	$C = \infty$, $\epsilon = 10^{-4}$, $Loss = quadratic$ and $KernelOptions = 0.5$
16	SVR	$C = \infty$, $\epsilon = 10^{-4}$
17	KRG	Quadratic regression, Gaussian correlation
18	KRG	Linear regression, exponential correlation
19	KRG	Constant regression, Gaussian correlation
20	KRG	Constant regression, exponential correlation
21	KRG	Linear regression, spline correlation
21	KRG	Constant regression, spline correlation
23	KRG	Linear regression, cubic correlation
24	KRG	Quadratic regression, cubic correlation
25	KRG	Constant regression, spherical correlation
26	KRG	Quadratic regression, spherical correlation
27	KRG	Constant regression, linear correlation
28	KRG	Quadratic regression, linear correlation
29	RBNN	$Spread = 1$ and $MN = N/5$
30	RBNN	$Spread = 1$ and $MN = N/10$

Obs.1: All other parameters not mentioned are kept with default values.
Obs.2: $\bar{y}$, σ_y and N are respectively: mean and standard deviation of y and number of sampling points.
Obs.3: For all KRG models: $\theta_0 = 10$ and $10^{-2} \leq \theta_i \leq 200$.
Obs.4: For all RBNN models: $Goal = (0.05\bar{y})^2$.
Obs.5: No attempt has been made in order to fine tuning the surrogates shape parameters.

Table 5 Summary of ensemble methods used as comparison in the present work. See details in Section 3.

Acronym	**Description and Equation**
SA	Simple Average, Eq. (3): $\hat{y}_{ens}(w_i, \mathbf{x}) = w_0 + \sum_{i=1}^{M} w_i \hat{y}_i(\mathbf{x})$, with $w_0 = 0$ and $w_i = \frac{1}{M}$.
PWS	PRESS Weighted Surrogate, Eq. (8): $w_i^{PWS} = \frac{w_i^*}{\sum_{j=1}^{M} w_j^*}$, with $w_i^* = (E_i + \alpha E_{avg})^\beta$ and $E_{avg} = \frac{1}{M}\sum_{j=1}^{M} E_j$.
minRMSE	Optimization of RMSE, Eq. (10): $\min_{w_i} \; E_{rr}(\hat{y}_{ens}, y)$, s.t. $\sum_{j=1}^{M} w_j = 1$, with $E_{rr}(\hat{y}_{ens}, y) = RMSE$.
OWS	Optimal Weighted Surrogate, Eq. (11): $\min_{w_i} \; \mathrm{MSE}(\hat{y}_{ens}) = \mathbf{w}^T\mathbf{C}\mathbf{w}$, s.t. $\sum_{j=1}^{M} w_j = 1$.

5.3 Results and Discussion

5.3.1 Effect of the Number of Augmenting Points

First of all, we investigated the effect of number of augmenting points, or the rate η_{aug}, in the augmented least squares approach. We selected the functions: Branin-Hoo ($n_v = 2$) to represent low dimension problems; and the function Ext. Rosenbrock ($n_v = 9$) to represent high dimension problems. We compared the average value and the standard deviation of the RMSE (in 100 runs) between the ordinary least squares solution (LS) and the augmented least squares approach (LS-a) with $M = 4$ metamodels in the ensembles (details in Tab. 3). In all cases, the size of the training data set (N_{tr}) was remained fixed as described in Tab. 2, and only the additional points (N_{add}) were varied to generate different rates η_{aug}.

The results are presented in Fig. 2. As we expected, it can be observed a reduction in the average and standard deviation of RMSE by increasing the number of augmenting points, for both low and high dimension problems. In case of low dimension problems (Branin-Hoo function), the most significant reduction in the levels of mean and standard deviation of RMSE is in the range $10\% < \eta_{aug} < 35\%$, and the behavior remained almost stable for $\eta_{aug} > 35\%$. That is, no improvement in RMSE was observed for $\eta_{aug} > 35\%$, for the low dimension problems. On the other hand, for high dimension problem (Ext. Rosenbrock function), no significant improvement in RMSE was observed for $\eta_{aug} > 10\%$. We observed a similar behavior for other functions as well, as showed in Fig.3.

It is important to observe that, in practice, the total number of sampling points, N, is in general fixed due to limitations on computer resources. In the present comparison, we decided to keep N_{tr} fixed for some reasons.

First of all, the motivation behind our LS-a method is *gathering additional data*, in order to improve the ordinary LS method, by reducing multicollinearity. We verified, after previous numerical experiments (see App. B), that is very difficult to solve multicollinearity in standard LS esnsemble only by adding more sampling points, specially for higher number of variables, due to the high density of points required. In this sense, we pursued other variants of LS method and we devised the LS-a ensemble as well to overcome this difficulty.

In the present comparative study, we aimed to verify how much one have "to pay" with additional points (N_{add}) and improve the predictability of LS ensemble, for a fixed number of training points (N_{tr}).

If, on the other hand, we keep N fixed and reduce the number of training points (N_{tr}), with different augmenting rates η_{aug}, then the predictability of N_{tr} tends to be reduced for a lower number of points, and the beneficial effects of the remaining out-of-sample points N_{add} should be canceled. In this way, the comparison would be confounding or even invalid for our purposes, since it is not possible to separate these two conflicting effects, i.e.: (i) reducing prediction quality (average RMSE) with lower number of training points N_{tr} and (ii) improving stability by augmenting points N_{add} (standard deviation of RMSE), simultaneously.

In order to illustrate this behavior we run the experiment by keeping N fixed and reducing the number of training points (N_{tr}), with different augmenting rates η_{aug} for Branin-Hoo ($n_v = 2$)Ext. Rosenbrock ($n_v = 9$) and the results are presented in Fig. 4. Observe that the prediction quality (average RMSE) tend to be lost with lower number of training points, with $\eta_{aug} > 30\%$.

Based on that, it is suggested to use augmentation points to generate the ensembles based on the rule $10\% < \eta_{aug} < 30\%$. In other words, if it is difficult or impossible to increase the data, then the available N sampling points should be split in two sets N_{tr} and N_{add}, with 10% to 35% of the data left aside to estimate the ensemble weights, by means of the augmented least squares approach.

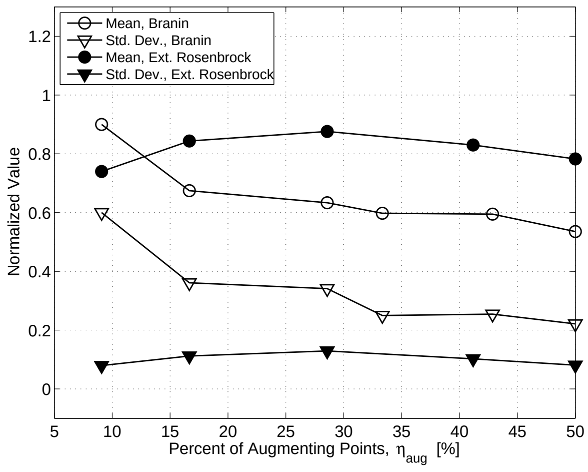
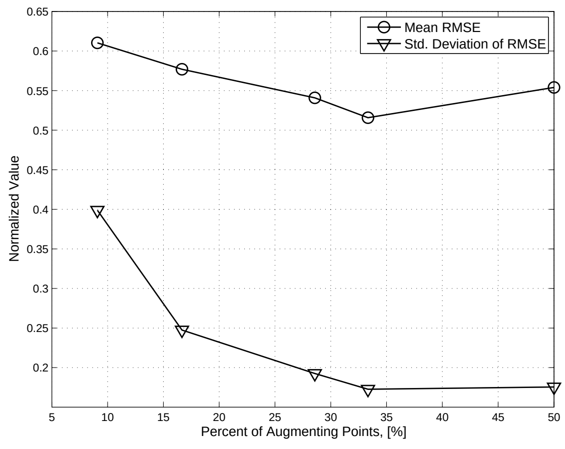
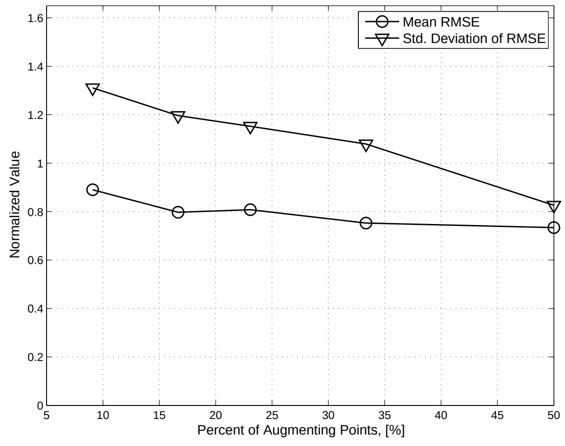


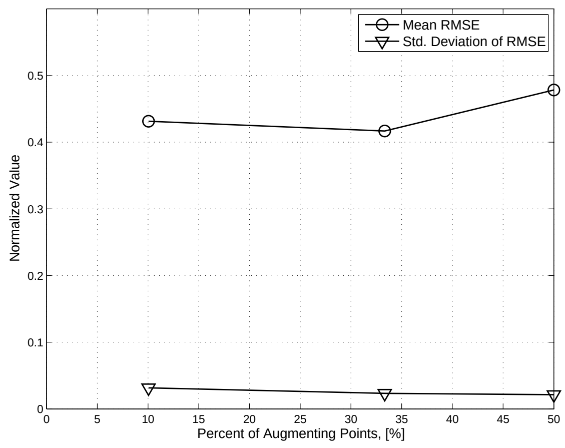
Fig. 2 Effect of number of augmenting points in the augmented least squares approach (LS-a). The mean value and standard deviation of RMSE in 100 runs are presented versus the rate η_{aug}, for the functions Branin-Hoo ($n_v = 2$) and Ext. Rosenbrock ($n_v = 9$). The results are normalized with respect to the standard least squares solution (LS) in each case. In all cases, the number of metamodels in the ensemble is $M = 4$.



(a) Hartman 3, $n_v = 3$



(b) Hartman 6, $n_v = 6$



(c) Dixon-Price, $n_v = 12$

Fig. 3 Effect of number of augmenting points in the augmented least squares approach (LS-a). The mean value and standard deviation of RMSE in 100 runs are presented versus the rate η_{aug}, for the functions Hartman ($n_v = 3$), Hartman ($n_v = 6$) and Dixon-Price ($n_v = 12$). The results are normalized with respect to the standard least squares solution (LS) in each case. In all cases, the number of metamodels in the ensemble is $M = 4$.

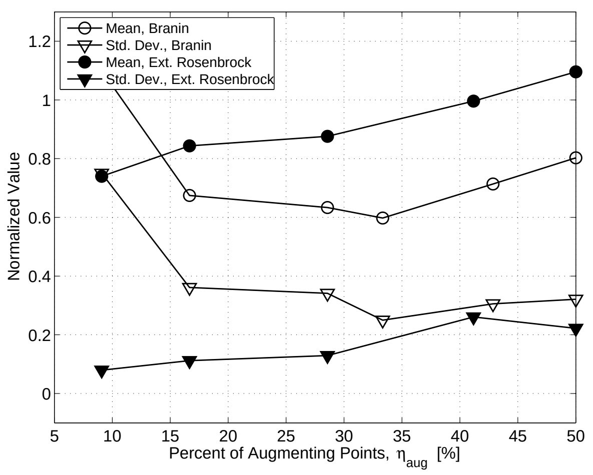


Fig. 4 Effect of number of augmenting points in the augmented least squares approach (LS-a), by keeping N fixed and reducing the number of training points (N_{tr}), with different augmenting rates η_{aug}, for the functions Branin-Hoo ($n_v = 2$) and Ext. Rosenbrock ($n_v = 9$). The mean value and standard deviation of RMSE in 100 runs are presented versus the rate η_{aug}. The results are normalized with respect to the standard least squares solution (LS) in each case. In all cases, the number of metamodels in the ensemble is $M = 4$.

5.3.2 Effect of Number of Metamodels

In a second test set, we verified the effect the number of metamodels in the ensemble quality of fit, in order to understand if the theoretical predictions of Eq. (6) apply in our case.

We took advantage of SURROGATES Toolbox flexibility to generate a large set of metamodels, in a similar way as presented in Viana et al (2009). Starting from the basic set ($M = 4$) presented in Tab. 3, we varied the tuning parameters and shape functions to create sets of different metamodels. For example: in case of PRS, we considered linear polynomials, instead of quadratic; In case of KRG, we changed the regression function to linear and the correlation from exponential to gaussian, etc. and so on, all the way up to ($M = 30$) metamodels, by using different instances of PRS, KRG, RBNN and SVR.See Tab. 4 for the metamodels using for $M > 4$.

The results for this test set are presented in Fig. 5. For low dimension problems it was observed a reduction on the error level of LS methods by doubling the number of metamodels in the ensemble (i.e., from 4 to 8), but this reduction on the average RMSE and standard deviation did not improved significantly from 8 to 30 metamodels in the ensemble. In case of high dimension problems the reduction of error levels of LS methods was not significant by increasing the number of metamodels in the ensemble. Again, we also observed

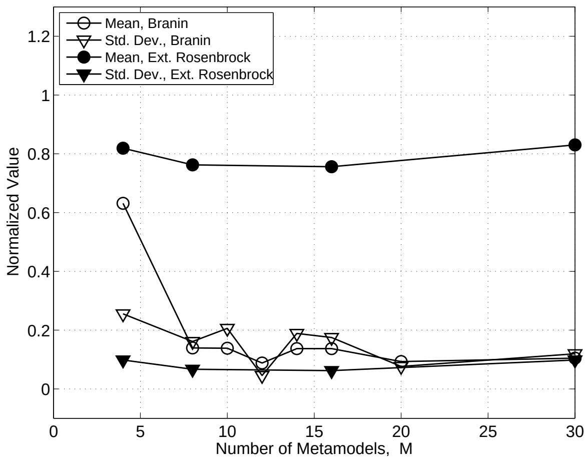


Fig. 5 Effect of number of metamodels in the ensemble by augmented least squares approach (LS-a). The mean value and standard deviation of RMSE in 100 runs are presented versus the number of metamodels M, for the functions Branin-Hoo ($n_v = 2$) and Ext. Rosenbrock ($n_v = 9$). The results are normalized with respect to the standard least squares solution (LS) in each case. In all cases, the rate of augmenting points is $\eta_{aug} = 33\%$.

a similar behavior for other test functions as well and the results are presented in Fig. 6.

In summary, based on these results, we did not confirmed the theoretical predictions based on Eq. (6), i.e., "the higher the number of distinct models, the lower is expected the MSE for the weighted average ensemble". In our experiments, the "ideal number of metamodels stayed at $M < 15$.

These results suggest that the typical problems of engineering applications do not meet the assumption of uncorrelated errors with zero mean and, in addition the underlying multicollinearity among models, specially as M increases, compromises the accuracy of the final ensemble prediction, and the expected reduction in MSE from Eq. (6) is not observed at all. In fact, the prediction quality decreased after $M > 15$, for some examples investigated, specially for large dimension problems.

5.3.3 Comparison of Least Squares Variants

In this test set we compared the performance of the augmented least squares LS-a and the other LS variants. The objective is to verify if it is possible to improve the LS solution by using methods suited to handle multicollinearity.

It can be observed based on Fig. 7 that most of the variants are able to reduce the average and standard deviation of RMSE with respect to the ordinary least squares (LS). See for example, in case of low dimension (Branin-Hoo), LS-a with normalized average RMSE $\mu = 0.58$ (or 42% reduction) and normalized standard deviation $\sigma = 0.28$ (or 72% reduction) with respect to LS, in 100 runs. For the majority of the other methods these reductions are around 30% for average value and 70% for standard deviation of RMSE. In case of high dimension problems (Ext. Rosenbrock) the reductions are also observed but they are more pronounced on standard deviation ($\approx 90\%$) than found for the average value of RMSE ($\approx 20\%$).

In all the cases studied, the best reduction in the average and standard deviation of RMSE was achieved by using the augmented approach LS-a ($\eta_{aug} = 33\%$). It can be observed that, even by combining the other LS variants with the augmented approach, we did not found any significant improvement in terms of average value and standard deviation of RMSE.

See for instance in Fig. 7 the similarity on the boxplots (and also on μ and σ values) for LS-a and the other variants by using the augmented approach (i.e., S-a, C-a, R-a, P-a, W-a, Ro-a and T-a). This trend has been confirmed with all the test problems investigated. See for example in Fig. 8 the same behavior found for the functions Hartman, $n_v = 3$; Hartman, $n_v = 6$ and Dixon-Price, $n_v = 12$.

Therefore, these results suggest that it does not worth to use different LS variants to reduce the error level and variance of ordinary LS in the creation of ensemble models. The highest improvement can be achieved by using the augmented method LS-a, with a proper number of augmenting points.

5.3.4 Effect of the Intercept

In this test set the aim is to verify the observations of Hashem (1993) regarding the effect of the intercept term in the ensemble.

In the problems studied, see Fig. 9, it was observed that the intercept term can be in fact not null in several runs for different DOE. Although it can be observed a high spread on the data, the average/median value for w_0 is around zero. Therefore, on a *local* perspective, the weights for the metamodels in the ensemble should be different, if the intercept term is considered or not, depending on the data (DOE).

On the other hand, on a *global* perspective, it was not observed in 100 runs any remarkable improvement in the average levels or standard deviation of RMSE for LS solutions by letting the intercept term to be nonzero in the ensemble. For instance, note the similarity on the boxplots of LS-a (without intercept) and LS-a-0 (with intercept) in the boxplots of Fig. 10. The same trend can be observed for the other LS methods (S, R, P, W, T and Ro), in low and high dimensions.

So, it seems to be a controversy on the local and global perspectives. Then, it is recommended to check for each problem and data (DOE) if the error levels can be improved or not by using the intercept term in the ensemble equation. Since in most cases the ensemble calculation is fast, specially when compared to the evaluation of the true models, then this verification should be worthwhile.

5.3.5 Comparison of Ensemble Methods

In this test set we compared the augmented least squares approach (LS-a, $\eta_{aug} = 33\%$) with other ensemble methods available in the literature. Refer to Tab. 5.

The methods OWS and PWS are available at SURROGATES Toolbox and we implemented the routines for SA and minRMSE. In all cases, we considered $M = 4$, as in Tab. 3 and data sets as in Tab. 2. The cross-validation procedure was performed with $k = 10$ in case of OWS and PWS.

As can be observed in the boxplots of Fig. 11, except for PWS, the performance of all ensemble methods was similar, in terms of average value and standard deviation of RMSE, for the functions Branin-Hoo, $n_v = 2$; Hartman, $n_v = 6$; and Dixon-Price, $n_v = 12$. On the other hand, in the same way as observed in Viana et al (2009), none of the ensemble methods performed significantly better than the best model in the set, in each run, in terms of RMSE (labeled as BestRMSE).

In these analytical cases investigated, the reduction on the average value of RMSE was lower than 3% and, as can be observed by the numbers $\mu(\sigma)$ in Fig. 11. In addition, it is observed that the standard deviation is equal or worst than BestRMSE, in most cases. The exception was in Dixon-Price function, but the reductions found on the standard deviation are not higher than 5%.

This same trend has been observed for "real-world" engineering problems (see Figs. 12 and 13). In the appendix (Sec. A.2) these simulation models currently applied in automotive industry are described.

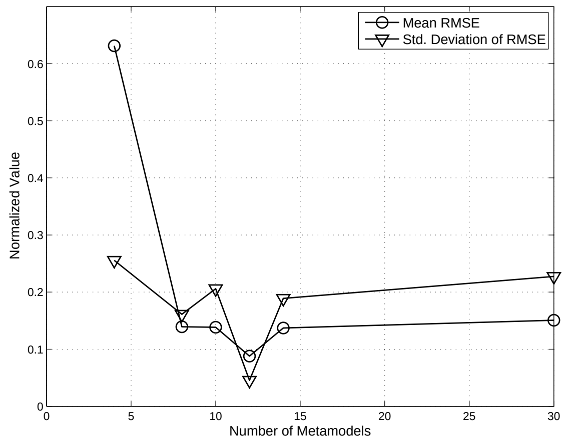
Although it can be observed up to 15% reduction in the average level of RMSE and up to 27% on the standard deviation for LS-a in truck responses ($n_v = 12$, see Fig. 12), the same was not observed in higher dimensions (i.e., car responses ($n_v = 30$ and $n_v = 44$), Fig. 13). In addition, both SA and PWS have shown to be inaccurate and unstable for the problems investigated. For the other ensemble methods, the performance was similar, in terms of average value and standard deviation of RMSE.

In terms of computational cost, for the studied problems the LS approach performed up to three orders of magnitude faster than the PRESS based methods, and in general it is comparable in terms of computational cost to the fastest method (i.e., the simple averaging ensemble, SA). See these results in Fig. 14.

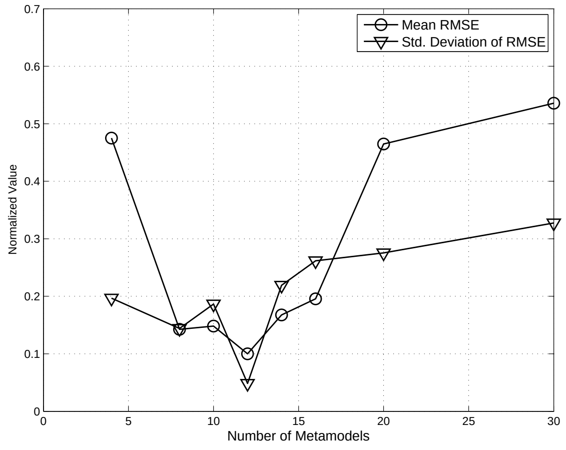
Finally, in terms of accuracy of prediction, we are forced to agree with the discussion by Viana et al (2009) and Viana (2011), with respect to multiple surrogates or ensemble of metamodels. Unfortunately, we did not find enough arguments to share the optimism of the related research with ensemble methods in data mining and machine learning, as for instance in Seni and Elder (2010), Zhou (2012) and Zhang and Ma (2012), in terms of prediction accuracy.

It seems that, due to the nature of the metamodeling, specially to deterministic problems, the multicollinearity will be always present, since the models tends to be similar among each other. In this way, finding a fair and accurate model tends to be the *selection* of the best model in a set (for instance, BestRMSE), instead of *combining* multiple models.

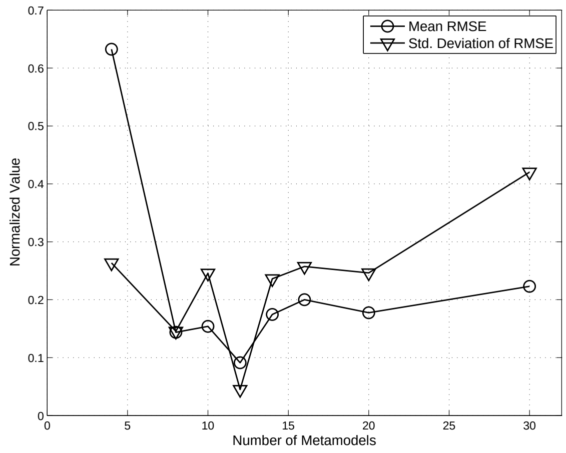
On the other hand, one hope for ensemble of metamodels usability should be in the efficient global optimization (EGO), as explored in the work Viana et al (2013). We also explored this front in the context of LS ensembles and the results are promising. The developments and results in this branch of application for the proposed ensemble approach will be presented the continuation of the present research in Ferreira and Serpa (2015b).



(a) Hartman 3, $n_v = 3$

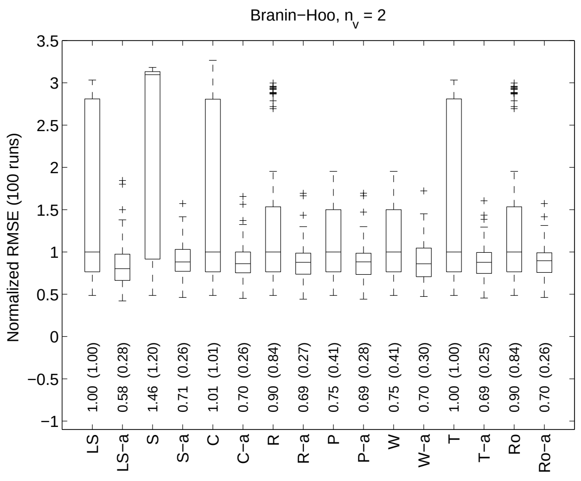


(b) Hartman 6, $n_v = 6$



(c) Dixon-Price, $n_v = 12$

Fig. 6 Effect of number of metamodels in the ensemble by augmented least squares approach (LS-a). The mean value and standard deviation of RMSE in 100 runs are presented versus the number of metamodels M, for the functions Hartman ($n_v = 3$), Hartman ($n_v = 6$) and Dixon-Price ($n_v = 12$). The results are normalized with respect to the standard least squares solution (LS) in each case. In all cases, the rate of augmenting points is $\eta_{aug} = 33\%$.



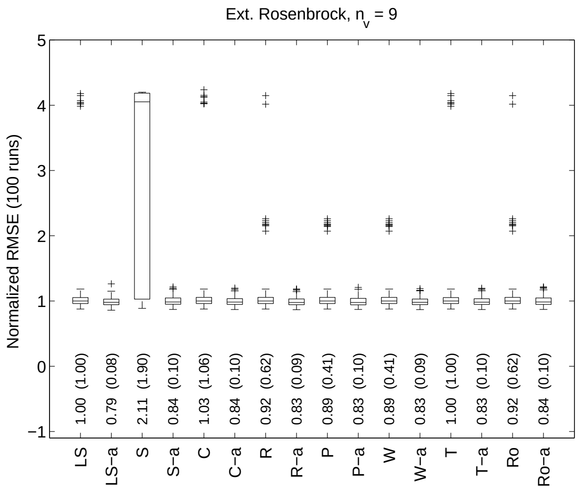
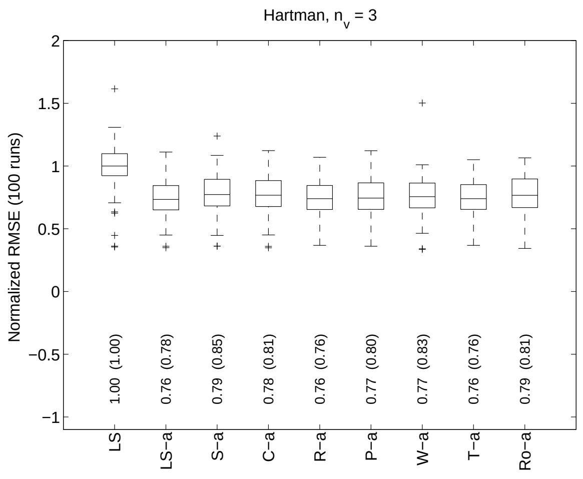
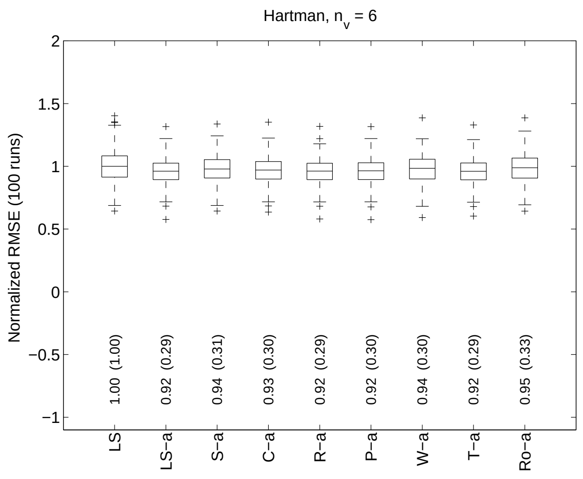


Fig. 7 Comparison of least squares variants and the effect of augmenting points for the functions Branin-Hoo ($n_v = 2$) and Ext. Rosenbrock ($n_v = 9$). Values in the bottom part of the graphs indicate respectively mean and standard deviation μ (σ) of RMSE for each method in 100 runs. All values normalized with respect to standard LS in each run. Suffix "-a" indicate the augmented method with $\eta_{aug} = 33\%$. In all cases, the number of metamodels in the ensemble is $M = 4$.





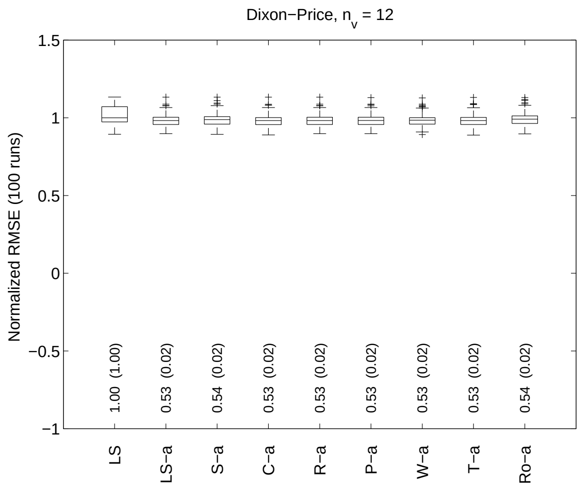
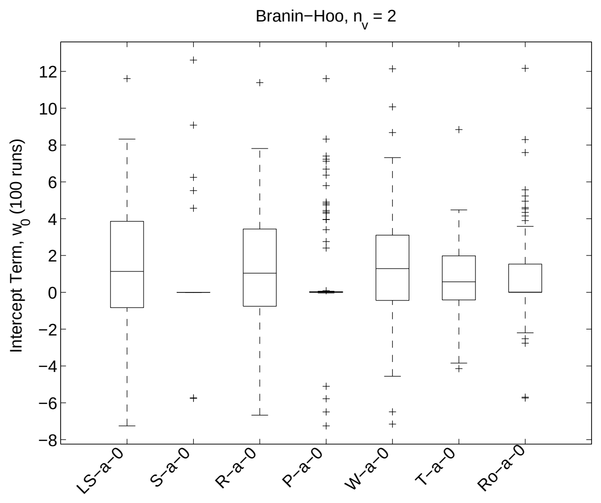


Fig. 8 Comparison of least squares variants and the effect of augmenting points for the functions Hartman ($n_v = 3$), Hartman ($n_v = 6$) and Dixon-Price ($n_v = 12$). Values in the bottom part of the graphs indicate respectively mean and standard deviation μ (σ) of RMSE for each method in 100 runs. All values normalized with respect to standard LS in each run. Suffix "-a" indicate the augmented method with $\eta_{aug} = 33\%$. In all cases, the number of metamodels in the ensemble is $M = 4$.



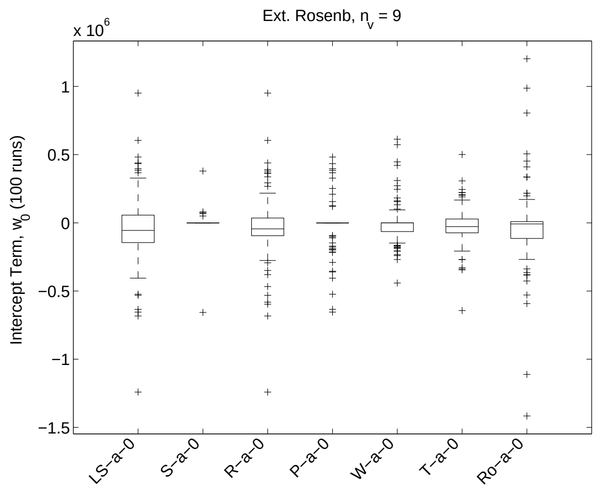
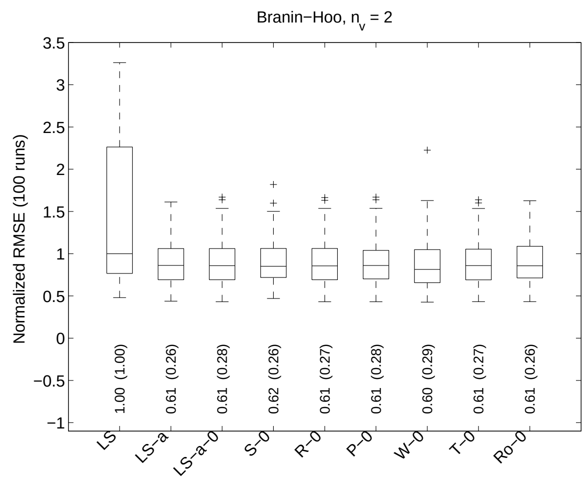


Fig. 9 Boxplots for the intercept term w_0 in 100 runs for the functions Branin-Hoo ($n_v = 2$) and Ext. Rosenbrock ($n_v = 9$). Suffix "-a" indicate the augmented method with $\eta_{aug} = 33\%$. Suffix "-0" indicate the use of the intercept w_0 in the ensemble. In all cases, the number of metamodels in the ensemble is $M = 4$.



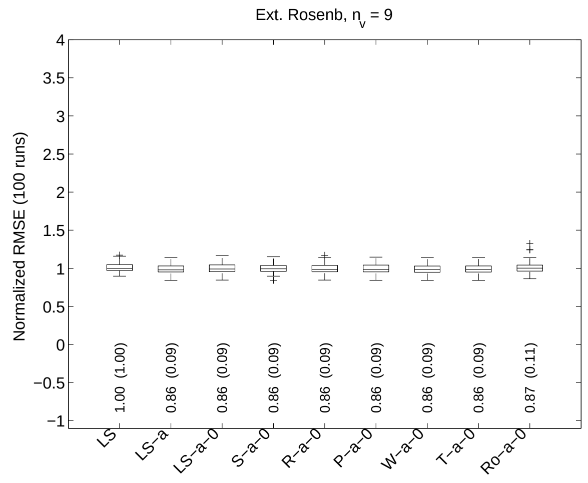
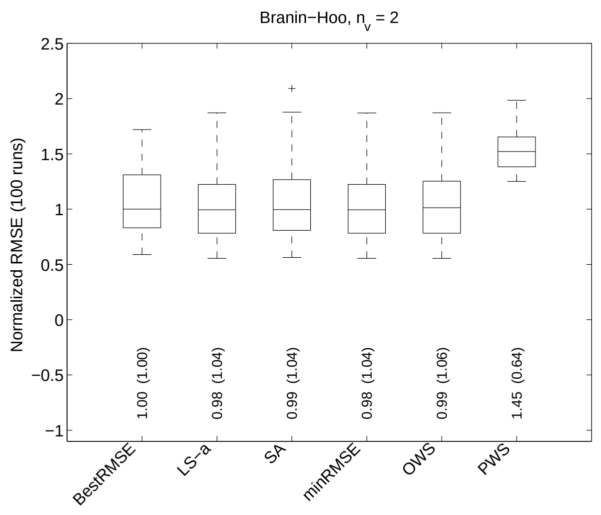
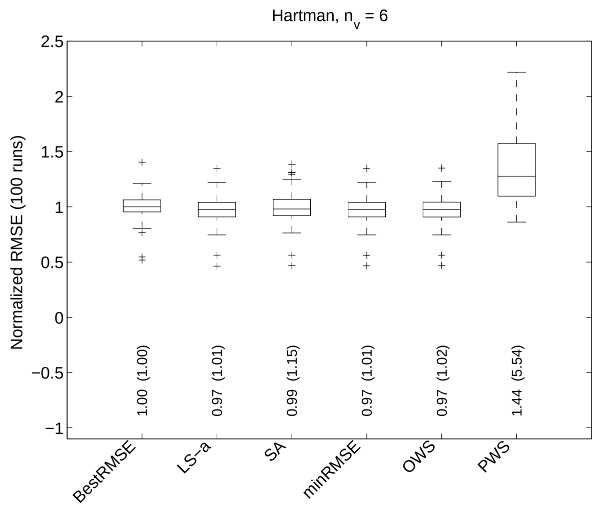


Fig. 10 Effect of the intercept term w_0 in the ensemble for the functions Branin-Hoo ($n_v = 2$) and Ext. Rosenbrock ($n_v = 9$). Values in the bottom part of the graphs indicate respectively mean and standard deviation μ (σ) of RMSE for each method in 100 runs. All values normalized with respect to standard LS in each run. Suffix "-a" indicate the augmented method with η_{aug} = 33%. Suffix "-0" indicate the use of the intercept w_0 in the ensemble. In all cases, the number of metamodels in the ensemble is $M = 4$.





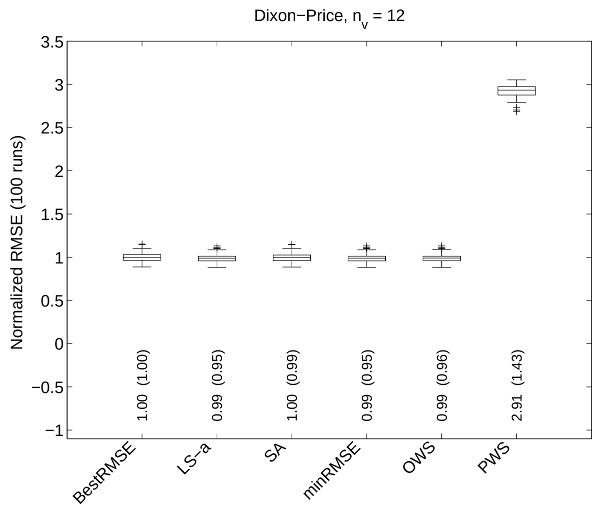


Fig. 11 Comparison among the augmented least squares (LS-a, η_{aug} = 33%) and other ensemble methods for the functions Branin-Hoo ($n_v = 2$), Hartman ($n_v = 6$) and Dixon-Price ($n_v = 12$). Values in the bottom part of the graphs indicate respectively mean and standard deviation μ (σ) of RMSE for each method in 100 runs. All values normalized with respect to BestRMSE in each run. In all cases, the number of metamodels in the ensemble is $M = 4$.

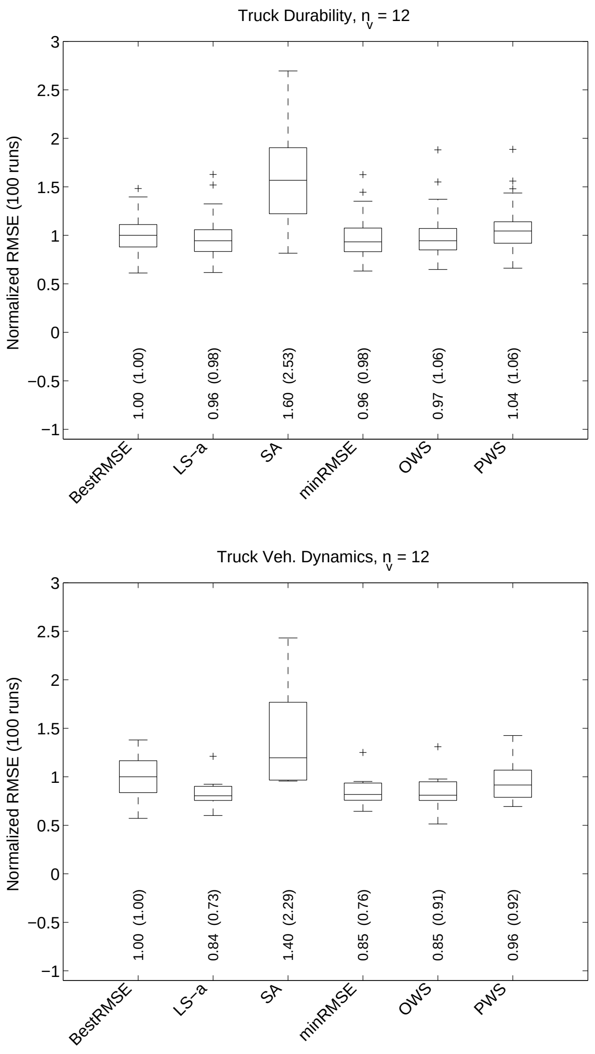


Fig. 12 Comparison among the augmented least squares (LS-a, $\eta_{aug} = 33\%$) and other ensemble methods for the truck responses with $n_v = 12$ variables. Values in the bottom part of the graphs indicate respectively mean and standard deviation μ (σ) of RMSE for each method in 100 runs. All values normalized with respect to BestRMSE in each run. In all cases, the number of metamodels in the ensemble is $M = 4$.

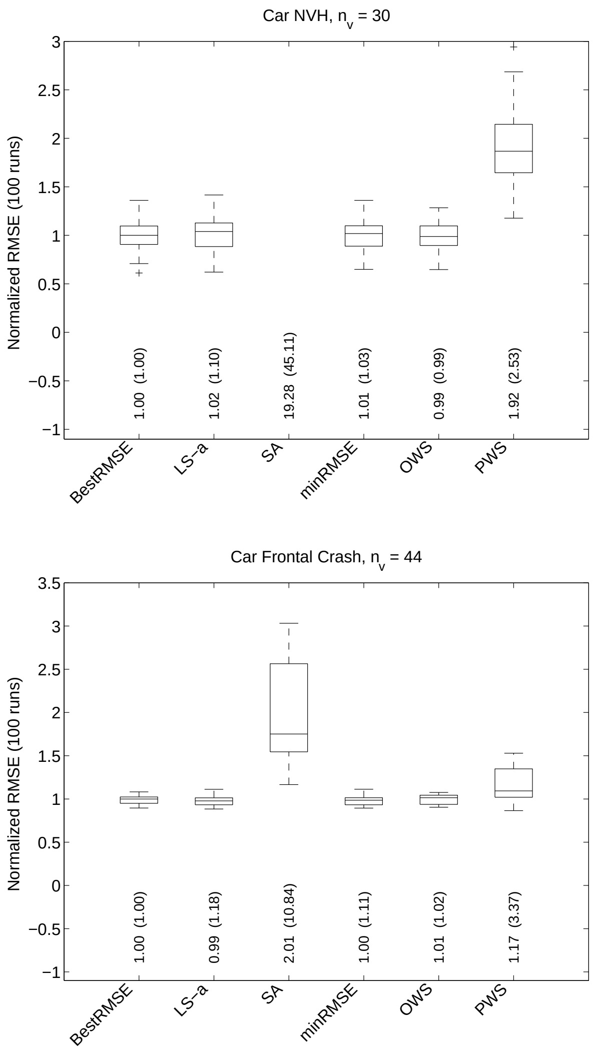
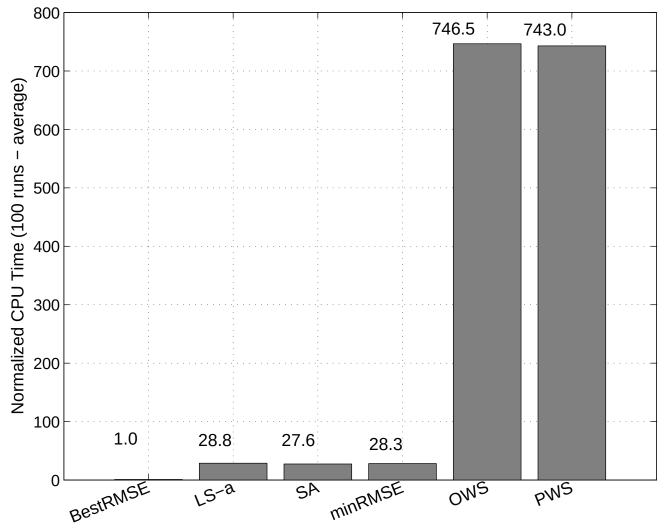
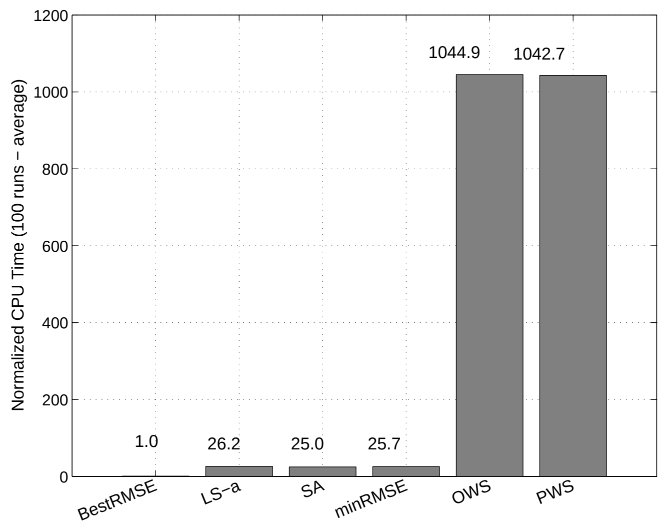


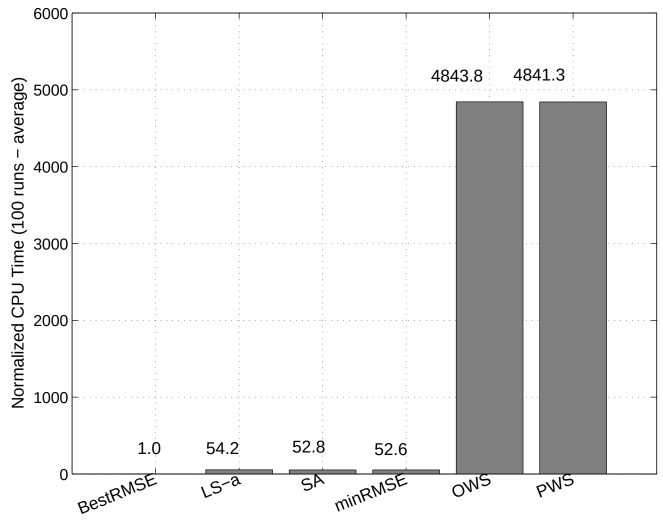
Fig. 13 Comparison among the augmented least squares (LS-a, $\eta_{aug} = 33\%$) and other ensemble methods for the car responses: Car NVH, $n_v = 30$, and Car Crash, $n_v = 44$ variables. Values in the bottom part of the graphs indicate respectively mean and standard deviation μ (σ) of RMSE for each method in 100 runs. All values normalized with respect to BestRMSE in each run. In all cases, the number of metamodels in the ensemble is $M = 4$.



(a) Truck Durability, $n_v = 12$



(b) Car NVH, $n_v = 30$



(c) Car Frontal Crash, $n_v = 44$

Fig. 14 Comparison of computational time among the augmented least squares (LS-a, $\eta_{aug} = 33\%$) and different ensemble methods, for engineering test problems. All values normalized with respect to BestRMSE in each run. In all cases, the number of metamodels in the ensemble is $M = 4$.

6 Concluding Remarks

In this work we presented an approach to create ensemble of metamodels (or weighted averaged surrogates) based on least squares (LS) approximation. The LS approach is appealing since it is possible to estimate the ensemble weights with simple formulation and low computational cost, without using any explicit error metric like PRESS (prediction sum of squares), as in most of the existent ensemble methods published in the literature.

The proposed LS approach is a variation of the standard least squares regression by augmenting the matrix system in such a way that reduces the effects of multicollinearity, inherent to calculation of the ensemble weights.

We investigated the number of augmenting points needed by the augmented LS method, in order to be effective. In summary, we observed that for low dimension problems the ideal number of data points to augment the system is around one third (30% to 35% of the data), and in case of high dimension this number is about 10% of the sampling points.

We also investigated the effect of increasing the number of metamodels in the augmented least squares ensemble. In summary, we did not confirmed the theoretical predictions that state: "the higher the number of distinct models, the lower is expected the MSE for the weighted average ensemble". For low dimension problems we observed a reduction in error from four to eight models in the ensemble and the average error level remained constant up to thirty metamodels. On the other hand, in case of high dimensions, the average error level remained almost constant by increasing the number of metamodels in the ensemble, from four to thirty models. These results suggest that the typical problems of engineering applications do not meet the assumption of uncorrelated errors with zero mean and, in addition the underlying multicollinearity among models compromises the accuracy of the final ensemble prediction.

In the sequence, we tested and compared the augmented LS approach with different LS variants and also with the existent ensemble methods, by means of analytical benchmark functions and real-world applications, in problems in the range of two to forty-four variables.

In comparison with existent LS variants (i.e., stepwise, ridge, principal components, constrained, weighted and total least squares), The augmented LS approach was able to reduce level of prediction error (average RMSE) the instability (standard deviation of RMSE) due to multicollinearity. For the test problems investigated and by using th 33% of the data as augmenting points, none of the other existent LS variants was able to surpass the performance of the proposed augmented LS approach, in terms of prediction error level (average RMSE) and stability (standard deviation of RMSE).

In the cases investigated, it was not observed any improvement in the error levels or variation of LS solution by letting the intercept term to be nonzero in the ensemble. Possibly there is a balance in the value of the remaining weights in the ensemble in order to compensate the presence of the intercept. On the other hand, since the intercept term is not necessarily null, it is recommended to check for each problem and data (DOE) if the error levels can be improved or not by using the intercept term in the ensemble equation. Since in most cases the ensemble calculation is fast, specially when compared to the evaluation of the true models, then this verification should be worthwhile.

When compared with other weighted average ensemble schemes published in the literature (i.e, simple averaging, PWS, direct optimization of RMSE and OWS methods), in general the augmented LS approach performed with good accuracy and stability for prediction purposes, in the same level of the existent ensemble methods. In terms of computational cost, for the problems studied, the LS approach performed up to three orders of magnitude faster than the PRESS based methods, and in general it has computational cost comparable to the fastest method (i.e., the simple averaging ensemble, SA).

As we discussed previously, an additional feature (nonexistent to the other existent ensemble methods) is that the ensemble of metamodels based on least squares has a prediction variance estimate function, that enables the application in the efficient global optimization context. The developments and results in this branch of application for the proposed ensemble approach will be presented the continuation of the present research, i.e., Ferreira and Serpa (2015b).

Acknowledgements The authors would like to thank Dr. F.A.C. Viana for the prompt help with SURROGATES Toolbox and also for the useful comments and discussions about the preliminary results of this work.

W.G. Ferreira would like to thank Ford Motor Company and also the support of his colleagues at MDO group and Product Development department that helped on the development of this work, which is part of his doctoral research underway at UNICAMP.

Finally, the authors are grateful for the comments and questions from the journal editor and reviewers. Undoubtedly their valuable suggestions helped to improve the clarity and consistency of the present text.

A Test Problems

A.1 Analytical Benchmarks

These functions were chosen since they are widely used to validate both metamodeling and optimization methods, as for example in Goel et al (2007), Acar and Rais-Rohani (2009) and Viana et al (2009).

Branin-Hoo

$$y(\mathbf{x}) = \left(x_2 + \frac{5.1x_1^2}{4\pi^2} + \frac{5x_1}{\pi} - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos(x_1) + 10, \quad (18)$$

for the region $-5 \leq x_1 \leq 10$ and $0 \leq x_2 \leq 15$.

Hartman

$$y(\mathbf{x}) = -\sum_{i=1}^{4} c_i \exp\left[-\sum_{j=1}^{n_v} a_{ij}\left(x_j - p_{ij}\right)^2\right], \quad (19)$$

where $x_i \in [0,1]^{n_v}$, with constants c_i, a_{ij} and p_{ij} given in Table 6, for the case $n_v = 3$ (Hartman-3); and in Table 7 Table 8, for the case $n_v = 6$ (Hartman-6).

Table 6 Data for Hartman-3 function.

i	c_i	a_{ij}			p_{ij}		
		$j=1$	2	3	$j=1$	2	3
1	1	3	10	30	0.3689	0.117	0.2673
2	1.2	0.1	10	35	0.4699	0.4387	0.747
3	3	3	10	30	0.1091	0.8732	0.5547
4	3.2	0.1	10	35	0.03815	0.5743	0.8828

Table 7 Data for Hartman-6 function, c_i and a_{ij}.

i	c_i	a_{ij}					
		$j=1$	2	3	4	5	6
1	1	10	3	1	3.5	1.7	8
2	1.2	0.05	10	17	0.1	8	14
3	3	3	3.5	1.7	10	17	8
4	3.2	17	8	0.05	10	0.1	14

Table 8 Data for Hartman-6 function, p_{ij}.

i	p_{ij}					
	$j=1$	2	3	4	5	6
1	0.1312	0.1696	0.5569	0.0124	0.8283	0.5886
2	0.2329	0.4135	0.8307	0.3736	0.1004	0.9991
3	0.2348	0.1451	0.3522	0.2883	0.3047	0.665
4	0.4047	0.8828	0.8732	0.5743	0.1091	0.0381

Extended Rosenbrock

$$y(\mathbf{x}) = \sum_{i=1}^{n_v-1}\left[(1-x_i)^2 + 100\left(x_{i+1} - x_i^2\right)^2\right], \quad (20)$$

where $x_i \in [-5,10]^{n_v}$.

Dixon-Price

$$y(\mathbf{x}) = (x_1 - 1)^2 + \sum_{i=2}^{n_v} i\left(2x_i^2 - x_{i-1}\right)^2, \quad (21)$$

where $x_i \in [-10,10]^{n_v}$.

Giunta-Watson

$$f(\mathbf{x}) = \sum_{i=1}^{n_v}\left[\frac{3}{10} + \sin\left(\frac{16}{15}x_i - 1\right) + \sin^2\left(\frac{16}{15}x_i - 1\right)\right], \quad (22)$$

where $\mathbf{x} \in [-2,4]^{n_v}$. This function the "noise-free" version of the function used in Giunta and Watson (1998).

A.2 Engineering Applications

In Figs. 15 and 16 are presented simulation models currently applied in automotive industry. These models are typical examples of the ones used in the Multidisciplinary Optimization (MDO) department at Ford Motor Company, where the first author of the present research works as structural optimization engineer. The examples described in this section are taken only as illustrations and they were part of a MDO project presented in a restrict conference summarized in the report by Ferreira et al (2012).

A regular MDO study at early design phases can comprise several models with hundreds of design variables and response functions to be monitored. After design sensitivity analysis stage, the top most significant variables and functions are selected in each model for metamodeling and multidisciplinary optimization.

We will use in this work the data available for the following variables and responses regarding these models to compare the performance of the ensemble methods discussed in this work by means of real-world applications.

Truck Models and Responses

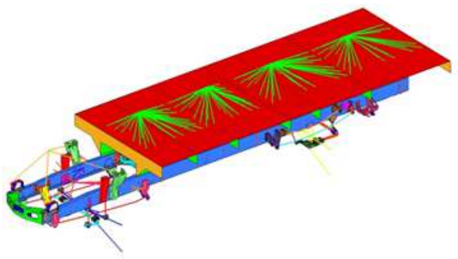
a) Truck Durability: it is presented in Fig. 15(a) a finite elements (FEM) model build in NASTRAN for truck frame durability evaluation. The durability responses (i.e., stress and/or fatigue/endurance metrics) are described as function of $n_v = 12$ geometry variables;

b) Truck Dynamics: it is presented in Fig. 15(b) a multibody model build in ADAMS for vehicle dynamics evaluation. The dynamics responses (i.e., displacements, velocities or accelerations for ride and handling performance) are defined based on the same $n_v = 12$ geometry variables used in the durability responses.

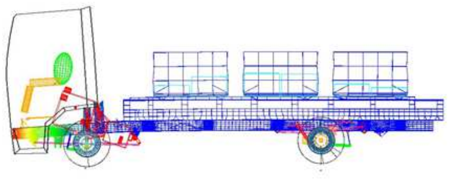
Car Models and Responses

a) Car NVH: it is presented in Fig. 16(a) a FEM model build in NASTRAN for passenger car NVH (*noise, vibration and harshness*) evaluation. The NVH response is described as function of $n_v = 30$ geometry variables;

b) Car Crash: it is presented in Fig. 16(b) a FEM model in RADIOSS for passenger car Frontal Crash evaluation. The crash responses (i.e., displacements, velocities or accelerations for safety performance) are described with $n_v = 44$ variables, that is the same 30 geometry used for NVH and additional 14 material parameters.



(a) Durability



(b) Vehicle Dynamics

Fig. 15 Examples of truck models applied in automotive industry for metamodeling and optimization. Courtesy of Ford Motor Company.

(a) NVH

(b) Frontal Crash

Fig. 16 Examples of passenger car models applied in automotive industry for metamodeling and optimization. Courtesy of Ford Motor Company.

B Preliminary Numerical Study

<**R1C3**>, <**R2C1**> and <**R2C3**>.

Our preliminary numerical experiments with LS ensembles were recorded in an internal research report at DMC-FEM-Unicamp. Since these results were not published before, we summarize the main findings here in this appendix for convenience.

B.1 Numerical Experiments Setup

We compared the performance of least squares ensemble (LS) with PRESS based methods, i.e., variations of OWS, Eq. (11), implemented SURROGATES Toolbox, Viana (2009).

At first we investigated the accuraccy as the number of sampling points increases for the case of two variables ($n_v = 2$) of Giunta-Watson function, Eq. (22), in the design space $\chi = \mathbf{x} \in [-2, 4]^{n_v}$.

In addition, we compared the methods for increasing the number of variables, i.e., $n_v = 1$, 2, 5 and 10, also for Giunta-Watson function, in the same design space. In this case, the number of sampling points were chosen based on the rule $N = 20n_v$, in order to keep the same point density as the dimension increases.

In all the cases investigated, we repeat the experiments with 100 different sampling points (DOE), to average out the influence of random data points on the quality of fit. The DOE are created by using the Latin Hypercube MATLAB function `lhsdesign`, optimized with `maxmin` criterion with 1000 iterations.

The ensemble of metamodels were composed with 4 distinct models, that is: PRS, KRG, RBNN and SVR, by considering the same setup presented in Tab. 3.

In all the examples, a total of $N_{test} = 2000$ test points were considered to calculate $RMSE$, as defined in Eq. (1). The cross-validation procedure, Eq. (2), was applied with $k = 10$, to balance accuracy and computational cost for PRESS calculation in OWS method and variations.

B.2 Summary of Results

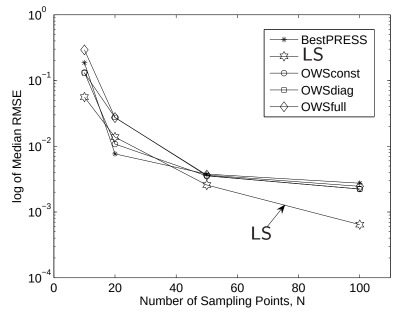
The main results of this preliminary study are compiled in Figure 17.

In summary, we observed that:

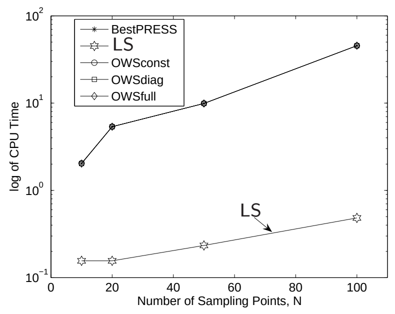
(i) The accuracy of LS ensemble method is on the same level of the PRESS based ensemble methods for moderate number of sampling points ($N < 50$) and LS was superior only for very dense design spaces, see Fig. 17(a). On the other hand, even for a very large number of sampling points, the computational cost for LS was always lower than OWS variants, i.e., more than one order of magnitude, see Fig. 17(b);

(ii) The accuracy of LS ensemble method is on the same level of the OWS ensemble methods, for increasing the number of variables, see Fig. 17(c). In the same way, the LS method performed much faster than OWS methods. At least one order of magnitude for low dimension problems (up to 5 variables) and more than two orders of magnitude for high dimension problems (10 variables), see Fig. 17(d);

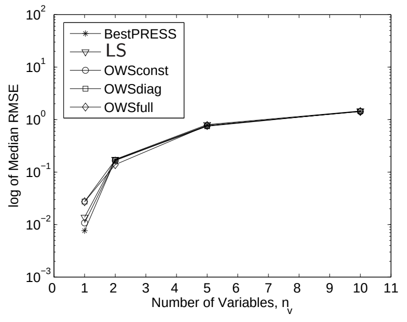
(iii) On the other hand, LS method presented an undesired instability (measured by the standard deviation of RMSE in 100 runs) as the number of variables increases, see Fig. 17(e);



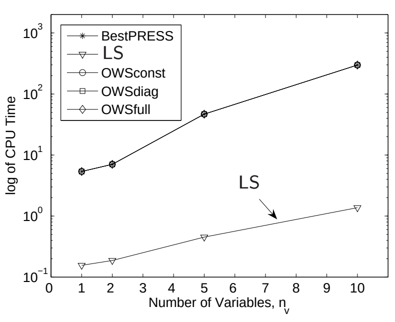
(a) $n_v = 2$



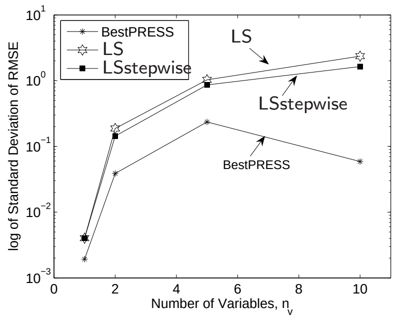
(b) $n_v = 2$



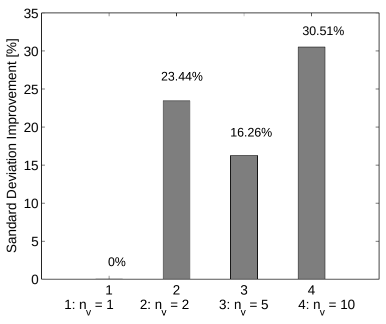
(c)



(d)



(e)



(f)

Fig. 17 Main results of a preliminary numerical study with least squares (LS) ensemble in comparison with PRESS based methods, i.e., variations of OWS.

(v) The variation of accuracy of LS method has been reduced around 30% for 10 variables by applying a *stepwise* selection procedure to the standard least squares solution, in order to reduce the effect of multicollinearity among the metamodels, see Fig. 17(f).

Based on these preliminary results, we concluded that the LS method can be viewed as an alternative of the PRESS based methods, since it is comparable in terms of accuracy and it performs much faster than the other ensemble methods available. In addition, the results with stepwise regression motivated a deeper investigation on the available methods for combating multicollinearity effects in least squares solution, in order to verify its feasibility for application in ensemble methods.

C Multicollinearity in Least Squares

<R1C1>

C.1 The Sources of Multicollinearity

The issue of multicollinearity in least squares regression is well known in statistics and related areas and the research in this front remounts at least to the decade of 1950. See for example Björk (1996) and Montgomery et al (2006) and the list of references therein for a broader perspective on this subject.

By definition, the least squares problem is based on the assumption that the k regressors x_i, or *predictor variables*, in the simple linear case of Eq. (12), are *mutually orthogonal.* In other words, it is assumed in advance that there is no linear relationship among the predictor variables.

In matrix form, the least squares problem can be stated as

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon} \ , \tag{23}$$

where $\mathbf{y}$ is a $(N \times 1)$ vector of responses; $\mathbf{X}$ is a $(N \times p)$ matrix of the regressor variables; $\boldsymbol{\beta}$ is a $(p \times 1)$ vector of unknown coefficients; and $\boldsymbol{\varepsilon}$ is a $(p \times 1)$ vector of random errors, that are assumed to be normally and independently distributed, with zero mean and finite variance, i.e. $\varepsilon_i \sim \text{NID}(0, \sigma^2)$. In this form, N represents the number of observations (or samples) and $p = k$ when the intercept term β_0 is considered zero and $p = (k + 1)$, otherwise.

One possible solution for Eq. (23) is the standard least squares estimator, i.e.,

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} \ , \tag{24}$$

that has the following properties:

(a) *Unbiasedness*:

$$\text{Bias}\left(\hat{\boldsymbol{\beta}}\right) \equiv E\left[\hat{\boldsymbol{\beta}}\right] - \boldsymbol{\beta} = 0 \quad \Rightarrow \quad E\left[\hat{\boldsymbol{\beta}}\right] = \boldsymbol{\beta} \ ; \tag{25}$$

(b) *Variance*:

$$\begin{aligned}\text{Var}\left(\hat{\boldsymbol{\beta}}\right) &\equiv E\left[\hat{\boldsymbol{\beta}}\hat{\boldsymbol{\beta}}^T\right] - E\left[\hat{\boldsymbol{\beta}}\right]\left(E\left[\hat{\boldsymbol{\beta}}\right]\right)^T \\ &= \sigma^2(\mathbf{X}^T\mathbf{X})^{-1} \ , \qquad \text{where} \\ \sigma^2 \approx \hat{\sigma}^2 &= \frac{\mathbf{y}^T\mathbf{y} - \hat{\boldsymbol{\beta}}^T\mathbf{X}^T\mathbf{y}}{N-p} \quad ;\end{aligned} \tag{26}$$

(c) *Mean Squared Error*:

$$\begin{aligned}\text{MSE}\left(\hat{\boldsymbol{\beta}}\right) &\equiv E\left[\left\|\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}\right\|^2\right] \\ &= \text{tr}\left\{\text{Var}\left(\hat{\boldsymbol{\beta}}\right)\right\} + \left\|\text{Bias}\left(\hat{\boldsymbol{\beta}}\right)\right\|^2 ;\end{aligned} \tag{27}$$

(d) *Gauss-Markov Theorem*: The least squares estimator $\hat{\boldsymbol{\beta}}$ is the best linear unbiased estimator (BLUE) of $\boldsymbol{\beta}$.

For proofs and details on these properties see Montgomery et al (2006) or Björk (1996).

Unfortunately, in most applications the assumption of mutually orthogonal does not hold and the final regression model can be misleading or erroneous. Thus, when there are linear or near-linear dependencies among the regressors, the problem of *multicollinearity* arises. This is due to the fact that the so called *correlation matrix* $\mathbf{X}^T\mathbf{X}$ has rank lower than p and for consequence the its inverse does not exist anymore. In this case, the least squares estimate $\hat{\boldsymbol{\beta}}$ becomes numerically unstable.

This instability of the coefficients can be explained by the variance definition. If the matrix $\mathbf{X}^T\mathbf{X}$ has linear dependence among columns (i.e., multicollinearity), then the variance of the coefficients can increase rapidly or become infinite and, by consequence the prediction will be poor.

Among the several sources of multicollinearity, the primary ones are: (i) the data collection method (size and distribution of sampling points); (ii) model overdefined or with redundant variables. During the decades, several methods have been devised for dealing with multicollinearity in least squares problems (see Montgomery et al (2006), Chap. 11). In general the techniques include gathering additional data and some kind of modification in the the way that the coefficients $\hat{\boldsymbol{\beta}}$ are estimated, in order to reduce the prediction errors induced by multicollinearity.

C.2 Prior Model Selection

In our context, the main source of multicollinearity is due to the fact the models $\hat{y}_i(\mathbf{x})$ tend to be very similar since all of them are trying to match the true response $y(\mathbf{x})$, as best as possible, therefore the problem is overdefined on its nature. In addition, this situation can be worsened if the sampling points are not well distributed in the design space.

In this way, by assuming that we have a fairly good design space distribution, the first option is conduct a prior selection and remove the most redundant models in the set $[\hat{y}_1(\mathbf{x}), \hat{y}_2(\mathbf{x}), \ldots, \hat{y}_M(\mathbf{x})]$, by means of some heuristic method.

One quick way, for instance, can be by using the concept of correlation, i.e. by defining the *pairwise correlation matrix* $\mathbf{R} = [r^2(\hat{y}_i, \hat{y}_j)]$, where $r(X_i, Y_i)$ is the *sample linear correlation coefficient*, that is,

$$r(X_i, Y_i) = \frac{\sum_{i=1}^{N}(X_i - \bar{X})(Y_i - \bar{Y})}{\left[\sum_{i=1}^{N}(X_i - \bar{X})^2 \sum_{i=1}^{N}(Y_i - \bar{Y})^2\right]^{\frac{1}{2}}} \tag{28}$$

with

$$\bar{X} = \frac{1}{N}\sum_{i=1}^{N} X_i,$$

for any two random vectors X_i and Y_i, of size N.

In this way, $R_{ii} = 1$ and $(0 \leq R_{ij} \leq 1)$, for $i \neq j$. Therefore, we can easily identify the most correlated models based on a threshold, say for example $(R_{ij} \geq 0.8)$, and verify if it is possible to eliminate the less significant model in the pair ij from the set, before creating the ensemble.

This heuristic approach can be useful when we have a large set of models, thus we can rapidly identify the most correlated pairs and remove the poorest ones in terms of accuracy in advance. It is worth noting that, specially for small sets, this criterion must be used carefully since interpolating or highly accurate models can lead to $(R_{ij} \rightarrow 1.0)$ as well, and of course cannot be discarded blindly.

Other diagnostics for multicollinearity exist in the least squares literature. Most of them are based on the examination of the correlation matrix $\mathbf{X}^T\mathbf{X}$, namely: correlation coefficients, determinant, eigensystem analysis, VIF (variance inflation factors), etc. We will not present them here, since at the end of the day all these diagnostic measures are useful to estimate pairwise correlation and not more than that. For example, it is possible to identify pairs of highly correlated models, but if the collinearity is among more than two models it cannot be identified by a simple inspection. In addition, as we already discussed, pure collinearity does not mean directly that models are not significant in terms of accuracy or predictability for the ensemble. Refer to Chap. 11 of Montgomery et al (2006) for a detailed discussion on this subject.

C.3 Gathering Additional Data

As reported in Montgomery et al (2006), one of the best methods to reduce the sources of multicollinearity in least squares is collecting additional data. The idea is first to understand the distribution of points in the design space and add more sampling points in the non-populated areas, in order to avoid concentrations along lines and therefore break-up multicollinearity.

In many cases, unfortunately, collecting additional data costly or even impossible. In other cases, collecting additional data is not a viable solution to the multicollinearity problem because its source is due to constraints on the model or in the population. This is the case of ensemble of metamodels. Since all the models $\hat{y}_i$ are trying to approximate the true response y as best as possible, or exactly in the case of interpolation, therefore the true response is acting y as a constraint in the problem, and then multicollinearity will come up naturally.

C.4 Variable Selection Methods in Regression

A central problem in least squares regression is related to the definition of the *best set* of variables or predictors to build the model. It is desired to define a *parsimonious model*, i.e. the simpler regression model that represent the problem at hand, as always as possible. A parsimonious model is easier to interpret, to collect data and, in addition, it is less prone to redundancies that induce linear dependencies and multicollinearity and reduce accuracy and predictability.

It is well known that the key driving question in all the variable selection methods is:
"How to include or exclude variables in a least squares model in order to achieve the desired accuracy and be parsimonious at same time?"

During the decades several methods have been devised, implemented and tested in an attempt to answer this question. In this sense, let us briefly explain the concept of balancing bias and variance in least squares approximation.

C.4.1 The Bias and Variance Dilemma

As remarked by Montgomery et al (2006), the Gauss-Markov property assures that the estimator $\hat{\boldsymbol{\beta}}$ has minimum error, in the least squares sense, among all unbiased linear estimators, but there is no guarantee that its variance will be small.

As we discussed previously, when the method of least squares is applied to nonorthogonal data, very inaccurate estimates of $\boldsymbol{\beta}$ can be obtained, due to the inflation of the variance. This implies that the absolute values of the coefficients are very unstable and may dramatically change in sign and magnitude by small variations in the design matrix $\mathbf{X}$.

One way to mitigate this issue is to relax the requirement that the estimator of $\boldsymbol{\beta}$ be unbiased. Let us assume that we can find a $\hat{\boldsymbol{\beta}}^*$ in such a way that

$$\hat{\boldsymbol{\beta}}^* = \hat{\boldsymbol{\beta}} - \boldsymbol{\delta}, \qquad \text{with} \qquad \|\boldsymbol{\delta}\| < \left\|\hat{\boldsymbol{\beta}}\right\|, \qquad \text{and} \qquad \mathrm{E}\left[\boldsymbol{\delta}\right] = \delta, \tag{29}$$

then the bias will be

$$\mathrm{Bias}\left(\hat{\boldsymbol{\beta}}^*\right) = \mathrm{E}\left[\hat{\boldsymbol{\beta}} - \boldsymbol{\delta}\right] - \boldsymbol{\beta} \quad \Rightarrow \quad \mathrm{Bias}\left(\hat{\boldsymbol{\beta}}^*\right) = -\delta \tag{30}$$

and, by assuming that $\hat{\boldsymbol{\beta}}$ and $\boldsymbol{\delta}$ are independent, then

$$\mathrm{Var}\left(\hat{\boldsymbol{\beta}}^*\right) = \mathrm{Var}\left(\hat{\boldsymbol{\beta}} - \boldsymbol{\delta}\right) = \mathrm{Var}\left(\hat{\boldsymbol{\beta}}\right) - \mathrm{Var}\left(\boldsymbol{\delta}\right). \tag{31}$$

In addition, the MSE will become

$$\begin{aligned}\mathrm{MSE}\left(\hat{\boldsymbol{\beta}}^*\right) &= \mathrm{tr}\left\{\mathrm{Var}\left(\hat{\boldsymbol{\beta}}\right)\right\} - \mathrm{tr}\left\{\mathrm{Var}\left(\boldsymbol{\delta}\right)\right\} + \|\boldsymbol{\delta}\|^2 \\ &= \mathrm{E}\left[\left\|\hat{\boldsymbol{\beta}} - \boldsymbol{\beta} - \boldsymbol{\delta}\right\|^2\right] \\ &= \mathrm{E}\left[2\left\|\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}\right\|^2 + 2\|\boldsymbol{\delta}\|^2 - \left\|\hat{\boldsymbol{\beta}} - \boldsymbol{\beta} + \boldsymbol{\delta}\right\|^2\right] \\ &= 2 \times \mathrm{MSE}\left(\hat{\boldsymbol{\beta}}\right) + f\left(\|\boldsymbol{\delta}\|^2\right)\end{aligned} \tag{32}$$

by using the parallelogram law for vector norms and the linearity of the expectation operator.

In summary, it can be concluded that, by allowing a small amount $\boldsymbol{\delta}$ of bias in $\hat{\boldsymbol{\beta}}^*$, the variance of $\hat{\boldsymbol{\beta}}^*$ will be smaller than $\hat{\boldsymbol{\beta}}$. On the other hand, the mean squared error at the data may increase rapidly as a function of the level of bias induced. If the effect of increasing bias is smaller than the effect of reducing variance then it is possible to reduce the error. Therefore, by controlling the size of $\hat{\boldsymbol{\beta}}$, it is possible to control the stability and error level of the solution by balancing bias and variance.

In Figure 18 is presented a geometrical interpretation on this behavior of the solution of $\hat{\boldsymbol{\beta}}$ in terms of bias and variance, for a generic problem with two variables, i.e. $\boldsymbol{\beta} = [\beta_1, \beta_2]^T$. The idea behind this behavior is by choosing a smaller estimator $\hat{\boldsymbol{\beta}}^*$, then the variance will be smaller. As consequence, the price for reducing variance will be always by adding bias in the solution, i.e. the mean squared error will not be the minimum anymore at the sampling points.

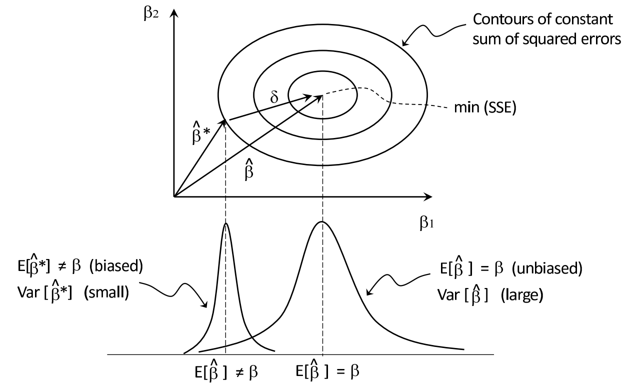


Fig. 18 An illustration of bias and variance dilemma in least squares. The BLUE estimator has minimal MS error but the associated variance is large. On the other hand, by accepting some bias or in other words *shrinking* the vector $\hat{\boldsymbol{\beta}}$, the MSE increases and the variance is reduced (improved stability). Adapted based on Montgomery et al (2006).

In practical terms, it means that if one chooses a biased estimator $\hat{\boldsymbol{\beta}}^*$, by increasing the mean squared error at sampling points, the variance will be reduced and the sensitivity of the regression coefficients to changes on the data will be also reduced (i.e. less sensitivity to noise or perturbations in the components of matrix $\mathbf{X}$). Therefore, with more stable coefficients, the overfitting can be reduced and the accuracy of predictions for future data will increase as well.

The possible ways to reduce the magnitude of the vector $\hat{\boldsymbol{\beta}}$ are mainly two: (i) by removing/combining variables from the scope of the model, or by forcing some of the $\hat{\beta}_i = 0$; and (ii) by reducing (shrinking) the size of the vector $\hat{\boldsymbol{\beta}}$.

Based on these two central ideas, most of the methods summarized in Sec. 4.3 were devised in order to find a solution on how to trade-off between bias and variance, and improve accuracy and predictability for a given set of variables in a problem least squares problem.

Finally, variable selection methods is a large front of research in least squares approximation field. Miller (2002) presented an extensive review on variable selection in regression problems and this is still a subject of active research, as can bee seen in the recent publications, for instance Ng (2012) which states that: "The variable selection is by no means solved." and Scheipl et al (2013) that reinforces that there is still a wide and open field for future research in variable and function selection in multivariate regression.

References

Acar E (2010) Various approaches for constructing an ensemble of metamodels using local error measures. Structural and Multidisciplinary Optimization 42(6):879–896

Acar E, Rais-Rohani M (2009) Ensemble of metamodels with optimized weight factors. Structural and Multidisciplinary Optimization 37(3):279–294

Akaike H (1974) A new look at the statistical model identification. IEEE Transactions on Automation and Control 19:716–723

Amemiya T (1985) Advanced Econometrics. Harvard University Pres, Cambridge, Massachsetts, USA

Bishop CM (1995) Neural Networks for Pattern Recognition. Oxford University Press Inc., New York, USA

Björk A (1996) Numerical Methods for Least Squares Problems. SIAM: Society for Industrial and Applied Mathematics

Breiman L (1996) Stacked regressions. Machine Learning 24:49–64

Efroymson MA (1960) Multiple regression analysis. In: Mathematical Methods for Digital Computers, Wiley, New York, USA, pp 191–203

Fan J, Li R (2001) Variable selection via nonconcave penalized likelihood and its oracle properties. Journal of American Statistical Association 96(456):1348–1360

Fang KT, Li R, Sudjianto A (2006) Design and Modeling for Computer Experiments. Computer Science and Data Analysis Series, Chapman & Hall/CRC, Boca Raton, USA

Ferreira WG, Serpa AL (2015b) Ensemble of metamodels: Extensions of the least squares approach to efficient global optimization. Structural and Multidisciplinary Optimization (submitted - ID SMO-15-0339)

Ferreira WG, Alves P, Slave R, Attrot W, Magalhaes M (2012) Optimization of a CLU truck frame. In: Ford Global Noise & Vibration Conference, Ford Motor Company, PUB-NVH108-02

Fierro RD, Bunch JR (1997) Regularization by truncated total least squares. SIAM Journal of Scientific Computation 18(4):1223–1241

Forrester A, Keane A (2009) Recent advances in surrogate-based optimization. Progress in Aerospace Sciences 45:50–79

Forrester A, Sóbester A, Keane A (2008) Engineering Desing Via Surrogate Modelling - A Practical Guide. John Wiley & Sons, United Kingdom

Foster DP, George EI (1994) The risk inflation criterion for multiple regression. Annals of Statistics 22:1947–1975

Giunta AA, Watson LT (1998) Comparison of approximation modeling techniques: polynomial versus interpolating models. In: 7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, AIAA-98-4758, pp 392–404

Goel T, Haftka RT, Shyy W, Queipo NV (2007) Ensemble of surrogates. Structural and Multidisciplinary Optimization 33:199–216

Golub GH, Heath M, Wahba G (1979) Generalizaed cross-validation as a method for choosing a good ridge parameter. Technometrics 21(2):215–223

Gunn SR (1997) Support vector machines for classification and regression. Technical Report. Image, Speech and Inteligent Systems Research Group. University of Southhampton, UK

Hannan EJ, Quinn BG (1979) The determination of the order of autoregression. Journal of Royal Statistics Society - Series B 41:190–195

Hashem S (1993) Optimal linear combinations of neural networks. PhD thesis, School of Industrial Engineering. Purdue University, West Lafayette, IN, USA

Hoerl AE, Kennard RW (1970a) Ridge regression: Applications to nonorthogonal problems. Technometrics 12(1):69–82

Hoerl AE, Kennard RW (1970b) Ridge regression: Biased estimation for nonorthogonal problems. Technometrics 12(1):55–67

Huber PJ, Rochetti EM (2009) Robust Statistics. Wiley Series in Probability and Statistics, John Wiley & Sons, Hoboken, New Jersey

van Huffel S, Vandewalle J (1991) The Total Least Squares Problem: Computational Aspects and Analysis. SIAM, Philadelphia, USA

Jekabsons G (2009) RBF: Radial basis function interpolation for matlab/octave. Riga Technical University, Latvia, version 1.1 ed.

Jolliffe IT (2002) Principal Component Analysis. Springer Series in Statistics, Springer, New York, USA

Jones DR, Schonlau M, Welch WJ (1998) Efficient global optimization of expensive black-box functions. Journal of Global Optimization 13:455–492

Koziel S, Leifesson L (2013) Surrogate-Based Modeling and Optimization - Applications in Engineering. Springer, New York, USA

Lai KK, Yu L, Wang SY, , Wei H (2006) A novel nonlinear neural network ensemble forecasting model for financial time series forecasting. In: Lecture Notes in Computer Science 3991, pp 790–793

Lophaven SN, Nielsen HB, Sondergaard J (2002) DACE - a matlab kriging toolbox. Tech. Rep. IMM-TR-2002-12, Technical University of Denmark

Markovsky I, van Huffel S (2007) Overview of total least-squares methods. Signal Processing 87:2283–2302

Meng C, Wu J (2012) A novel nonlinear neural network ensemble model using k-plsr for rainfall forecasting. In:

Bio-Inspired Computing Applications. Lecture Notes in Computer Science 6840, pp 41–48

Miller A (2002) Subset Selection in Regression. Monographs on Statistics and Applied Probability, Chapman & Hall/CRC, USA

Montgomery DC, Peck EA, Vining GG (2006) Introduction to Linear Regression Analysis. Wiley Series in Probability and Statistics, John Wiley & Sons, Hoboken, New Jersey

Ng S (2012) Variable selection in predictive regressions. In: Handbook of Economical Forecasting, Elsevier, pp 752–789

Perrone MP, Cooper LN (1993) When networks disagree: Ensemble methods for hybrid neural networks. Artificial Neural Networks for Speech and Vision, Chapman & Hall, London, UK

Queipo NV, et al (2005) Surrogate-based analysis and optimization. Progress in Aerospace Sciences 41:1–28

Ramu M, Prabhu RV (2013) Metamodel based analysis and its applications: A review. Acta Technica Corviniensis - Bulletin of Engineering 4(2):25–34

Rasmussen CE, Williams CK (2006) Gaussian Processes for Machine Learning. The MIT Press

Rousseeuw PJ, Leroy AM (2003) Robust Regression and Outlier Detection. Wiley Series in Probability and Statistics, John Wiley & Sons, Hoboken, New Jersey

Sanchez E, Pintos S, Queipo NV (2008) Toward and optimal ensemble of kernel-based approximations with engineering applications. Structural and Multidisciplinary Optimization 36:247–261

Scheipl F, Kneib T, Fahrmeir L (2013) Penalized likelihood and bayesian function selection in regression models. Advances in Statistical Analysis 97(4):349–385

Schwarz G (1978) Estimating the dimension of a model. Annals of Statistics 6:461–464

Seni G, Elder J (2010) Ensemble Methods in Data Mining: Improving Accuracy Through Combining Predictions. Synthesis Lectures on Data Mining and Knowledge Discovery, Morgan & Claypool Publishers, Chicago, IL, USA

Shibata R (1984) Approximation efficiency of a selection procedure for a number of regression variables. Biometrika 71:43–49

Simpson TW, Toropov V, Balabanov V, Viana FAC (2008) Design and analysis of computer experiments in multidisciplinary design optimization: A review of how far we have come - or not. In: 12th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Victoria, British Columbia

Thacker WI, Zhang J, Watson LT, Birch JB, Iyer MA, Berry MW (2010) Algorithm 905: SHEPPACK: modified shepard algorithm for interpolation of scattered multivariate data. ACM Transactions on Mathematical Software 37(3):1–20

Tibshirani R (1996) Regression shrinkage and selection via lasso. Journal of Royal Statistical Society 58(1):267–288

Viana FAC (2009) SURROGATES toolbox user's guide version 2.0 (release 3). Available at website: http://fchegury.googlepages.com

Viana FAC (2011) Multiples surrogates for prediction and optimization. PhD thesis, University of Florida, Gainesville, FL, USA

Viana FAC, Haftka RT, Steffen V (2009) Multiple surrogates: how cross-validation error can help us to obtain the best predictor. Structural and Multidisciplinary Optimization 39(4):439–457

Viana FAC, Cogu C, Haftka RT (2010) Making the most out of surrogate models: tricks of the trade. In: Proceedings of the ASME 2010 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference IDETC/CIE 2010, Montreal, Quebec, Canada

Viana FAC, Haftka RT, Watson LT (2013) Efficient global optimization algorithm assisted by multiple surrogates techniques. Journal of Global Optimization 56:669–689

Weisberg S (1985) Applied Linear Regression. Wiley Series in Probability and Statistics, John Wiley & Sons, New Jersey, USA

Wolpert D (1992) Stacked generalizations. Neural Networks 5:241–259

Yang XS, Koziel S, Liefsson L (2013) Computational optimization, modeling and simulation: Recent trends and challenges. Procedia Computer Science 18:855–860

Yu L, Wang SY, Lai KK (2005) A novel nonlinear ensemble forecasting model incorporating glar and ann for foreign exchange rates. Computers and Operations Research 32:2523–2541

Zerpa LE, Queipo NV, Pintos S, Salager JL (2005) An optimization methodology of alkaline-surfactant-polymer flooding processes using field scale numerical simulation and multiple surrogates. Journal of Petroleum Science and Engineering 47:197–208

Zhang C, Ma Y (2012) Ensemble Machine Learning. Methods and Applications. Springer, New York, USA

Zhou ZH (2012) Ensemble Methods. Foundations and Algorithms. Machine Learning & Pattern Recognition Series, Chapman & Hall/CRC, Boca Raton, USA

Structural and Multidisciplinary Optimization manuscript No.
(will be inserted by the editor)

Ensemble of metamodels: extensions of the least squares approach to efficient global optimization

Wallace G. Ferreira · Alberto L. Serpa

Received: date / Accepted: date

Abstract In this work we present LSEGO, an approach to drive efficient global optimization (EGO), based on LS (least squares) ensemble of metamodels. By means of LS ensemble of metamodels it is possible to estimate the uncertainty of the prediction with any kind of model (not only kriging) and provide an estimate for the expected improvement function. For the problems studied, the proposed LSEGO algorithm has shown to be able to find the global optimum with less number of optimization cycles than the required by the classical EGO approach. As more infill points are added per cycle, the faster is the convergence to the global optimum (exploitation) and also the quality improvement of the metamodel in the design space (exploration), specially as the number of variables increases, when the standard single point EGO can be quite slow to reach the optimum. LSEGO has shown to be a feasible option to drive EGO with ensemble of metamodels as well for constrained problems, and it is not restricted to kriging and to single infill point per optimization cycle.

Keywords Ensemble of metamodels · Surrogate based optimization · Efficient global optimization · Least squares

W. G. Ferreira
Ford Motor Company Brazil
CAE & Optimization Engineering
Av. Taboão, 899
09655-900, S. B. Campo, SP, Brazil
Tel.: +55-11-41744207
E-mail: wferrei7@ford.com - wgferreira@yahoo.com

A. L. Serpa
University of Campinas - UNICAMP
School of Mechanical Engineering - FEM
Department of Computational Mechanics - DMC
13083-970, Campinas, SP, Brazil
Tel.: +55-19-35213387
E-mail: serpa@fem.unicamp.br

1 Introduction

In the last decades, the use of metamodeling methods (also known as *surrogate modeling* or *response surface methodology*) to replace expensive computer simulation models such as FE (Finite Elements) or CFD (Computational Fluid Dynamics) found in automotive, aerospace and oil-gas industry, for example, has become a common place in both research and practice in engineering design, analysis and optimization. A collection of engineering research and applications in this field has been recently published in Koziel and Leifesson (2013).

In this context, surrogate based (or response surface based, or metamodel based) optimization refers to the process of using fast running metamodels as surrogates of original complex and long time running computer simulation models to approximate the objectives and constraint functions in a standard optimization algorithm. This methodology has shown to be effective in both multidisciplinary and multiobjective optimization problems and it has been widely applied in research and industry. Refer to the reviews by Queipo et al (2005), Simpson et al (2008) and Forrester and Keane (2009) for a broad and detailed discussion on this subject.

The surrogate based optimization is in general an iterative (cyclic) process. At each cycle, instances of simulation models with different parameters are evaluated (sampling points from a Design of Experiments, DOE). The surrogate models are fit based on these sampling data and the resulting approximate functions are used in the search of optimum points (exploitation), analysis of the response behavior, sensitivity and trends in the design space (exploration). Once optimum design candidates and other extrema points are found, they are evaluated with the true simulation models and if necessary the new points are included in the sampling space in order to improve the approximation and to restart the iterative process. In Jones (2001) it is presented a review on different approaches (also known as sequential sampling approaches) used to drive the global surrogate based optimization.

Efficient global optimization (EGO) is an iterative surrogate based approach that, at each optimization cycle, one *infill* point is selected as the one that maximizes the *expected improvement* with respect to the minimum of the objective function. The EGO concept emerged after the work of Jones et al (1998), that was based mainly on previous research on Bayesian Global Optimization (Schonlau (1997) and Mockus (1994)). Traditionally EGO algorithms are based on kriging approximation and a single infill point is generated per optimization cycle.

A question that arises is that the selection of only one infill point per optimization cycle can be quite slow to achieve the convergence to the optimum. If parallel computation is an available resource, then multiple infill points should be defined and therefore less cycles might be required for convergence. This aspect can be crucial specially if the computer models take several hours to run[1] and a single point EGO approach becomes prohibitive, specially in multidisciplinary optimization scenarios. In practical terms, when parallel resources are not an issue, it can be worthwhile to run more simulations per cycle in order to reach an optimum in a reasonable lead time, even if the total number of simulations is higher at the end of the optimization process.

The aspect of single versus multiple infill points per cycle is known and discussed since the origins of EGO-type algorithms at later 1990s. Schonlau (1997) proposed extensions to the standard EGO algorithm to deliver "m points per stage", but he pointed out numerical difficulties in the evaluation of the derived expressions accurately at a reasonable computer cost. In addition, his results were limited to a two variable function and indicate 18.9% penalty for the parallel approach in terms of function evaluations for the same accuracy.

Besides this apparent disadvantage in terms of function evaluations, it can be observed an increasing research interest on EGO approaches with multiple infill points per cycle published in the last ten years. This is probably because parallel computation is nowadays a relatively easy resource and the potential penalty of parallel EGO approaches should be neglected in favor of fast delivering results. See for example the works by Sóbester et al (2004), Henkenjohann and Kukert (2007), Ponweiser et al (2008), Ginsbourger et al (2010) and Viana et al (2013).

In our previous research work, Ferreira and Serpa (2015), we presented and discussed the concept of least squares (LS) regression, in order to find the optimal weights in ensemble of metamodels (or weighted average surrogates, WAS). We proposed the *augmented least squares ensemble of metamodels* (LS-a), a variation of the standard least squares regression to create ensemble of metamodels. In this way, the ensemble of metamodels constructed based on LS approach inherits the variance estimator, which can be used in the definition of the expected improvement function to be applied in EGO-type algorithms.

In the present work we extend the results of Ferreira and Serpa (2015) by proposing an approach that is able to provide multiple points per cycle in a EGO-type algorithm by using least squares ensemble of metamodels. We use the acronym LSEGO, that stands for "least squares ensemble efficient global optimization". In the numerical experiments performed, LSEGO has shown to be a feasible alternative to drive EGO algorithms with ensemble of metamodels, and in addition it is not restricted to kriging nor a single infill point per optimization cycle. LESEGO was applied in the optimization of analytical benchmark functions (up to six variables) and including some examples of constrained optimization as well.

The remainder of the present text will be divided as follows. In Section 2 we present the theoretical fundamentals of efficient global optimization and the standard EGO algorithm. In Section 3 we present and discuss our proposed LSEGO approach. In Section 4 we present the numerical experiments performed to validate and compare the LSEGO approach with the traditional EGO algorithm and the respective results and discussion are presented in Section 5. Finally, the concluding remarks are presented in Section 6.

2 Theoretical background

2.1 Metamodel based optimization

Let $\mathbf{x} = [x_1 \ \ldots \ x_{n_v}]^T \in \Re^{n_v}$ be a vector of n_v parameters or design variables. Metamodels are nothing but methods that attempt to fit a function $\hat{y} = \hat{f}(\mathbf{x}) : \Re^{n_v} \to \Re$ to a set of known N data points $\chi : \{\mathbf{x}_i, \ y_i\}$, determined by a sampling plan (design of experiments, DOE).

[1] For instance, even with high end computers clusters used nowadays in automotive industry, one single full vehicle analysis of high fidelity safety crash FEM model takes up to 15 processing hours with 48 CPU in parallel. With respect to CFD analysis one single complete car aerodynamics model for drag calculation, by using 96 CPU, should take up 30 hours to finish. An interesting essay regarding this "never-ending" need of computer resources in structural optimization can be found in Venkatararaman and Haftka (2004).

In most of the models, the approximate function $\hat{y}(\mathbf{x}) \approx y(\mathbf{x})$ (response surface, surrogate or metamodel) can be given in the general form

$$\hat{y}(\mathbf{x}) = \mathbf{w}^T \boldsymbol{\psi}(\mathbf{x}), \tag{1}$$

where $\mathbf{w}$ represents *weights* to be determined and $\boldsymbol{\psi}(\mathbf{x})$ are the associated *basis functions*.

Further details on formulation and different types of metamodels (e.g. polynomial response surfaces (PRS), radial basis functions (RBF), neural networks (NN), support vector regression (SVF), etc.) can be found in Forrester et al (2008), Fang et al (2006) and in the references therein. In the Appendix A it its presented the basic formulation for kriging metamodel (KRG), that will be referenced throughout this work.

The optimization process based on metamodels (i.e., metamodel based, response surface based or surrogate based optimization) can be stated as a standard optimization problem, by replacing the true objective function $y = f(\mathbf{x})$ and the n_c constraints $g_i(\mathbf{x})$, that are complex and costly to compute, by their respective fast and cheap surrogates, i.e.,

$$\mathbf{x}_{opt} = \begin{cases} \min\limits_{\mathbf{x}} \quad \hat{y}(\mathbf{x}) \\ \\ \text{s.t.,} \quad \hat{g}_i(\mathbf{x}) - g^i_{max} \quad \leq 0, \quad i = 1 \ldots n_c \\ \\ \qquad \mathbf{x}_{lb} \leq \mathbf{x} \leq \mathbf{x}_{ub} \end{cases}, \tag{2}$$

where g^i_{max} is the reference value (target) for i-th constraint[2] and $\mathbf{x}_{lb}$ and $\mathbf{x}_{ub}$ the bounds on the design space. The optimum point $\mathbf{x}_{opt}$ can be found by any available optimization algorithm: gradient based, direct search, genetic algorithm, etc. Refer to Queipo et al (2005) or Forrester and Keane (2009) for details.

2.2 Efficient global optimization (EGO)

As pointed out by Forrester and Keane (2009): "the Holy Grail of global optimization is finding the correct balance between exploitation and exploration". Algorithms that favor exploitation (search of the optimum), can be quite slow to converge and in addition be trapped at a local minimum point and not be able to reach the global optimum. On the other hand, pure exploration (improvement of the surrogate and search in the hole design space) may lead to a waste of resources (function evaluations, simulations). In this sense, efficient global optimization (EGO) algorithms emerge as feasible tools to balance exploitation and exploration of the design space.

[2] Only for notational convenience, without loss of generality, we will assume that all equality constraints $h(\mathbf{x})$ can be properly transformed into inequality constraints $g(\mathbf{x})$.

Since the data set χ is arbitrary, the determination of $\hat{y}(\mathbf{x})$ can be stated as a realization of an stochastic process. In this sense, the approximation can be modeled as Gaussian, i.e., normally distributed random variable $\hat{Y}(\mathbf{x})$, with mean $\hat{y}(\mathbf{x})$ and variance $\hat{s}^2(\mathbf{x})$.

Efficient global optimization algorithms are centered in the concept of maximum expected improvement. Let $y_{min} = \min(y_1, \ldots, y_n)$ be the best current value for the function $y(\mathbf{x})$ in the sampling data set. The *improvement* is defined as $I(\mathbf{x}) = \max(0, y_{min} - Y(\mathbf{x}))$. Then, the *probability of improvement* of a point $\mathbf{x}$ with respect to y_{min}, can be calculated as

$$P[I(\mathbf{x})] = \frac{1}{\hat{s}\sqrt{2\pi}} \int_{-\infty}^{0} \exp\left(\frac{-\left[I - \hat{y}\right]^2}{2\hat{s}^2} \right) \mathrm{d}I, \tag{3}$$

by abbreviating the dependence of I, $\hat{s}$ and $\hat{y}$ on $\mathbf{x}$.

On the other hand, the amount of improvement expected can be obtained by taking the expectation $E[\cdot]$ of $I(\mathbf{x})$, which leads to

$$E[I(\mathbf{x})] = \hat{I}(\mathbf{x}) \boldsymbol{\Phi}\left(\frac{\hat{I}(\mathbf{x})}{\hat{s}(\mathbf{x})} \right) + \hat{s}(\mathbf{x}) \phi\left(\frac{\hat{I}(\mathbf{x})}{\hat{s}(\mathbf{x})} \right), \tag{4}$$

for $s(\mathbf{x}) > 0$, and $E[I(\mathbf{x})] = 0$, otherwise. In this equation, $\hat{I}(\mathbf{x}) \equiv (y_{min} - \hat{y}(\mathbf{x}))$ and $\boldsymbol{\Phi}(\cdot)$ and $\phi(\cdot)$ are respectively the normal cumulative distribution and normal probability density functions.

Eq. 4 is known as the *expected improvement function* of any point $\mathbf{x}$ in the design space, with respect to the current best value y_{min}. For details in the derivation see Forrester and Keane (2009) or the original publications of Schonlau (1997) and Jones et al (1998).

By means of Eq. 4, the EGO algorithm can be defined for any metamodel $\hat{y}(\mathbf{x})$, that provides an uncertainty estimate $\hat{s}(\mathbf{x})$. EGO was originally devised and is traditionally applied with KRG (see basic formulation at Appendix A), but other models like PRS, RBF and other Gaussian models can be used as well. See for example Sóbester et al (2004) in which RBF was applied.

2.3 The standard EGO algorithm

The standard EGO algorithm can be summarized in the following steps:

i. Define a set χ of N sampling points and start the optimization cycles ($j \leftarrow 1$);
ii. Evaluate the true response $y(\mathbf{x})$ at all data sites in χ, at the current cycle N, and set

$$y_{min} \leftarrow \min(y_1, \cdots, y_N);$$

iii. Generate the metamodel $\hat{y}(\mathbf{x})$ and estimate $E[I(\mathbf{x})]$ with all the data points available at the current cycle N;
iv. Find the next infill point $\mathbf{x}_{N+j}$ as the maximizer of $E[I(\mathbf{x})]$; evaluate the true function $y(\mathbf{x})$ at $\mathbf{x}_{N+j}$ and add this new point to the sampling space χ;
v. If the stopping criteria is not met, set

$$y_{min} \leftarrow \min(y_1, \cdots, y_{N+j}),$$

update cycle counter $(j \leftarrow j+1)$ and return to step iii. Otherwise, finish the EGO algorithm.

2.4 Extensions for EGO algorithm

The EGO algorithm can be extended to handle constraints in the optimization, by using the concept of probability of improvement. The basis for this extension can be found in Schonlau (1997) and with details and applications in Forrester et al (2008) and Han and Zhang (2012).

By following the derivation presented in Han and Zhang (2012), the idea is to find the probability of satisfying the constraints $g_i(\mathbf{x})$. In other words, when $P[G_i(\mathbf{x}) \leq 0] \to 1$, the constraint is satisfied; otherwise, when $P[G_i(\mathbf{x}) \leq 0] \to 0$, the constraint is violated. Analogously to Eq. 3, $P[G_i(\mathbf{x}) \leq 0]$ can be calculated as

$$P[G_i(\mathbf{x}) \leq 0] = \frac{1}{\hat{s}_i\sqrt{2\pi}} \int_{-\infty}^{0} \exp\left(-\frac{[G_i - \hat{g}_i]^2}{2\hat{s}_i^2}\right) \mathrm{d}G_i \quad (5)$$

where $G_i(\mathbf{x})$ is the random variable related to $\hat{g}_i(\mathbf{x})$ and $\hat{s}_i(\mathbf{x})$ the respective constraint uncertainty estimate.

In this way, the step iv. of the EGO standard algorithm presented at the end of Section 2.2 can be modified as follows to accommodate n_c *independent* and *uncorrelated* constraints, i.e.,

$$\mathbf{x}_{N+j} = \begin{cases} \max_{\mathbf{x}} \quad E[I(\mathbf{x})] \times \prod_{i=1}^{n_c} P[G_i(\mathbf{x}) \leq 0] \\ \\ \text{s.t.,} \quad \mathbf{x}_{lb} \leq \mathbf{x} \leq \mathbf{x}_{ub} \end{cases} . \quad (6)$$

The treatment of multiobjective optimization problems can be also extended by using the concept of probability and expected improvement. It is required a more elaborated derivation that is out of the scope of this text and the details can be found in Forrester et al (2008). In addition, Jurecka (2007) has successfully extended and applied the EGO concept to treat robust optimization problems as well.

These and other possible extensions of EGO-type algorithms are of interest for research and practical applications and we intend to explore this field in our future work.

3 LS ensemble of metamodels EGO (LSEGO)

3.1 Definitions

As we presented in Ferreira and Serpa (2015), the linear ensemble of metamodels can be written as

$$\hat{y}_{ens} = \hat{\mathbf{Y}}\mathbf{w} \quad (7)$$

where $\mathbf{y} = [y_1 \ \cdots \ y_N]^T$, $\hat{\mathbf{Y}} = [\hat{y}_1(\mathbf{x}_i) \ \cdots \ \hat{y}_M(\mathbf{x}_i)]$ and $\mathbf{w} = [w_1 \ \cdots \ w_M]^T$, for N sampling points and M metamodels.

If the optimal weights $\mathbf{w}$ are estimated by least squares methods (LS), as we discussed in detail in Ferreira and Serpa (2015), then the resulting ensemble of metamodels inherits the *least squares variance estimate* $V[\hat{y}_{ens}(\mathbf{x})] \equiv \hat{s}^2(\mathbf{x})$, for the prediction at each point $\mathbf{x}$, that can be written as

$$\hat{s}^2(\mathbf{x}) = \hat{\sigma}^2 [\hat{\mathbf{y}}(\mathbf{x})]^T \left(\hat{\mathbf{Y}}^T\hat{\mathbf{Y}}\right)^{-1} \hat{\mathbf{y}}(\mathbf{x}) \ , \quad (8)$$

with $\hat{\mathbf{y}}(\mathbf{x}) = [\hat{y}_1(\mathbf{x}) \ \hat{y}_2(\mathbf{x}) \ \cdots \ \hat{y}_M(\mathbf{x})]^T$, and

$$\hat{\sigma}^2 = \frac{\hat{\mathbf{y}}_{ens}^T\hat{\mathbf{y}}_{ens} - \hat{\mathbf{w}}^T\hat{\mathbf{Y}}\hat{\mathbf{y}}_{ens}}{N - n_v} \quad (9)$$

where $\hat{\mathbf{y}}_{ens} = [\hat{y}_{ens}(\mathbf{x}_1) \ \hat{y}_{ens}(\mathbf{x}_2) \ \cdots \ \hat{y}_{ens}(\mathbf{x}_N)]^T$.

Therefore, if $\hat{s}^2(\mathbf{x})$ is a good estimate for the uncertainty of the LS ensemble of metamodel at the design space, then it can be used to derive the expected improvement function (Eq. 4) in order to drive EGO algorithms, by using any kind of metamodels $\hat{y}_i(\mathbf{x})$ and not only kriging.

In fact, we cannot verify or prove *a priori* that $\hat{s}^2(\mathbf{x})$ is good or not for our purposes in the EGO context. Since it is a well-established and accepted estimate for general least squares regression, it will be applied in the present work without any proof. The main assumption is that if the LS ensemble of metamodel has reasonable prediction accuracy, therefore the variance prediction $\hat{s}^2(\mathbf{x})$ should be reasonable as well. Our numerical experiments showed that $\hat{s}^2(\mathbf{x})$ works well for generating the EI function and the optimization is convergent in several problems investigated. These results and conclusions will be presented and discussed in the next sections.

In this sense, we named this proposed approach as LSEGO (least squares ensemble efficient global optimization) and the intention in the present work is to verify and demonstrate the efficiency of this method as an alternative to drive EGO-type algorithms.

3.2 Illustrations: one infill point per cycle

Fig. 1 illustrates the evolution of LSEGO for a one variable function with one infill point per optimization cycle. At each optimization cycle it is presented the true function $y(x)$ versus the approximation by LS-a ensemble (left plot) and the expected improvement function $E[I(x)]$ (right plot), calculated by means of Eq. 4, with $\hat{s}^2(\mathbf{x})$ as defined in Eq. 8.

At cycle 01 we have a very poor approximation with correlation coefficient $R^2 = 0.258$ and normalized root mean squared error $NRMSE = 31.9\%$. The $E[I(X)]$ presents a clearly defined peak close to the true minimum ($x_{opt}^{exact} = 5.624$). Note for this example in Fig. 1, at the first six optimization cycles, the maximum of expected improvement function works in the "exploitation mode" and prioritizes to add infill points around the optimum.

At this stage (cycle 06) the optimum found $x_{opt} = 5.600$ is quite close to the exact value (0.43% error), with a very good quality approximation for the metamodel: $R^2 = 0.975$ and $NRMSE = 4.2\%$. After cycle 06, the LSEGO algorithm automatically switches to the "exploration mode" and the infill points are selected in order to improve the quality of the approximation in the whole design space, instead of improving the minimum value found. The LSEGO algorithm was stopped at cycle 12 with $x_{opt} = 5.600$, $R^2 = 0.999$ and $NRMSE = 0.5\%$.

This behavior of LSEGO in one dimension was observed for other functions as well, with different levels of nonlinearity and multimodality. Based on these preliminary results we can conclude that LSEGO performed quite well in terms of exploitation and exploration of the design space.

On the other hand, for higher dimension problems we noted a very slow convergence for the algorithm, associated to a high concentration of infill points around the global optimum, as observed in standard EGO algorithm as well. See for instance in Fig. 2 the behavior for LSEGO for Giunta-Watson function (see Appendix B, Eq. 20) with two variables and one infill point per optimization cycle. The exact minimum is accurately found only at cycle 37 and all the infill points are located at this neighborhood. As consequence, the quality of the approximation at cycle 37 is still poor outside the optimum vicinity (note the difference on the exact and approximate contours for the function).

It is well known that the expected improvement function can be extremely multimodal (i.e., with mutiple peaks that lead to several locations with high probable improvement). This behavior was observed in most of the optimization cycles investigated for LSEGO as well (see for instance the $E[I(x)]$ curves in Fig. 1).

This multimodal behavior of the expected improvement function can be taken as advantage by selecting more than one infill point per optimization cycle in order to accelerate the whole optimization process, as discussed in Sóbester et al (2004), for example.

3.3 LSEGO with parallel infill points

As we discussed in the Introduction (Section 1) the aspect of single versus multiple infill points per cycle is known and discussed since the origins of EGO-type algorithms (Schonlau (1997) and Jones et al (1998)).

Sóbester et al (2004) used a multistart optimization algorithm to find multiple maximum points for the expected improvement function and take advantage of parallel processing resources. Their results indicated accelerated convergence for the optimization with significant reduction in processing time.

In the last years we can observe an increasing research interest on EGO approaches with multiple infill points per cycle. Henkenjohann and Kukert (2007), Ponweiser et al (2008) and Ginsbourger et al (2010), for instance, proposed different implementations of parallel EGO approaches by extending the concepts of *generalized expected improvement* and *m-step improvement* proposed in the work by Schonlau (1997).

In a different way, Viana et al (2013) proposed the MSEGO (multiple surrogates EGO). In this case, they used multiple surrogates simultaneously at each cycle of EGO algorithm. With at least one kriging model available, they imported the uncertainty estimate for this model to the other non-kriging models in the set. By means of different uncertainty estimates, they generate different instances of expected improvement functions to be maximized and to provide multiple parallel infill points in each EGO cycle.

In the same direction of the approach proposed by Viana et al (2013), in the present work we suggest an alternative scheme to generate multiple infill points per cycle in the LSEGO algorithm by taking advantage of multiple surrogates in a form of a LS ensemble.

Since we have an arbitrary set of M distinct metamodels, that are relatively fast to generate (as compared with the true simulation model $y(\mathbf{x})$), then it is possible to create and arbitrary number of N_p *partial* LS ensembles $\hat{y}_{ens}^k(\mathbf{x})$, by generating permutations of $\bar{M} < M$ metamodels. Therefore, by means of the N_p partial LS ensembles there are N_p respective expected improvement functions $E^k[I(\mathbf{x})]$ available to generate up to N_p infill points per cycle.

In this case, differently from Viana et al (2013), it is not required to have at least one kriging model in the set to generate the uncertainty estimates, since in LSEGO $\hat{s}_k^2(\mathbf{x})$ are generated directly from the least squares definition for the partial ensembles.

Based on preliminary tests we observed a good performance of LSEGO for the purpose of multiple infill points per cycle. We will illustrate this application of LSEGO at Section 3.4 and the final LSEGO algorithm will be summarized in Section 3.5.

3.4 Illustrations: multiple infill points per cycle

We will illustrate the behavior of LSEGO with multiple infill points with functions of two variables. Fig. 4 shows the same setup used in the case of Fig. 2 for Giunta-Watson function (2 variables), but now with $N_p = 8$ infill points per optimization cycle.

Note in Fig. 4 that, by allowing more infill points per cycle, LSEGO converges very quickly to the exact optimum at cycle 5 (as compared to cycle 37, for $N_p = 1$ in Fig. 2), with a reasonable correlation for the metamodel at this point ($R^2 = 0.803$ and $NRMSE = 8.1\%$). In addition, if we let LSEGO to continue with the exploration, the metamodel quality is continuously improved. Observe the results at cycle 12 ($R^2 = 0.998$ and $NRMSE = 0.6\%$), that can be explained by the more spread of infill points (exploration), not only on the vicinity of the optimum (exploitation), as in the case for one infill point per cycle.

We observed the same performance of LSEGO for other functions as well. See for instance the evolution for Branin-Hoo function (ref. App. B, Eq. 18) in Fig. 6. In this case, LSEGO has found the tree optimum points within high accuracy at cycle 05 ($R^2 = 0.999$ and $NRMSE = 0.5\%$).

Based on these preliminary results with one and two dimension functions, we can conclude that LSEGO has a good performance on driving EGO algorithm, with single and multiple infill points per cycle. As more infill points are added per cycle, faster is the convergence to the global optimum (exploitation) and also the quality improvement (predictability) of the metamodel in the whole design domain (exploration). In the Section 4 we will show the numerical experiments performed with the objective to compare the performance of the proposed algorithm LSEGO versus the traditional EGO.

3.5 LSEGO algorithm with parallel infill points

The LSEGO algorithm with parallel infill points can be summarized in the following steps:

i. Define a set χ of N sampling points and start the optimization cycles ($j \leftarrow 1$);

ii. Evaluate the true response $y(\mathbf{x})$ at all data sites in χ, at the current cycle N, and set

$$y_{min} \leftarrow \min(y_1, \cdots, y_N);$$

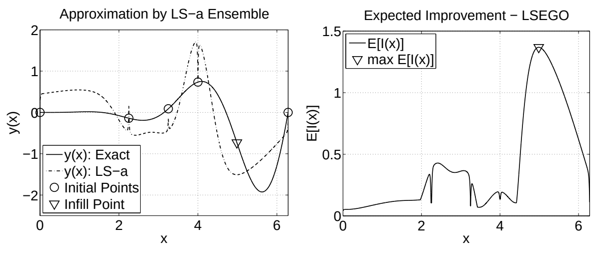
iii. Generate the M metamodels $\hat{y}_i(\mathbf{x})$, the N_p partial LS ensembles $\hat{y}^k_{ens}(\mathbf{x})$ and the respective expected improvement functions $E^k[I(\mathbf{x})]$, with all data available at the current cycle N;

iv. Find the set of next distinct $N_p^* \leq N_p$ infill points $\chi_{infill} = \left[\mathbf{x}_{N+j}, \cdots, \mathbf{x}_{N+j+N_p^*}\right]$ as the respective maximizers of $E^k[I(\mathbf{x})]$; evaluate the true function $y\left(\mathbf{x}\right)$ at the N_p^* infill points and add them to the sampling space: $\chi \leftarrow \chi \cup \chi_{infill}$;

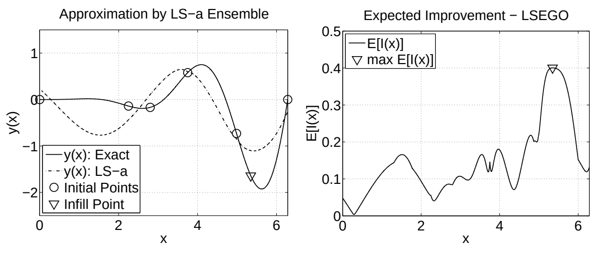
v. If the stopping criteria is not met, set

$$y_{min} \leftarrow \min(y_1, \cdots, y_{N+j}, \cdots, y_{N+j+N_p^*}),$$

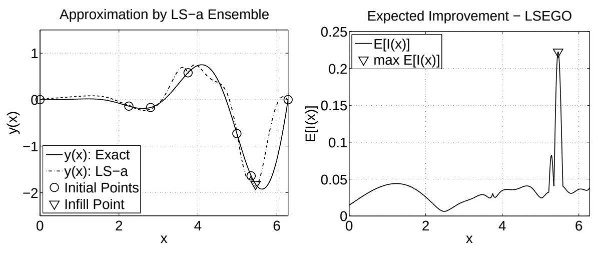
update cycle counter ($j \leftarrow j+1$) and return to step iii. Otherwise, finish the LSEGO algorithm.



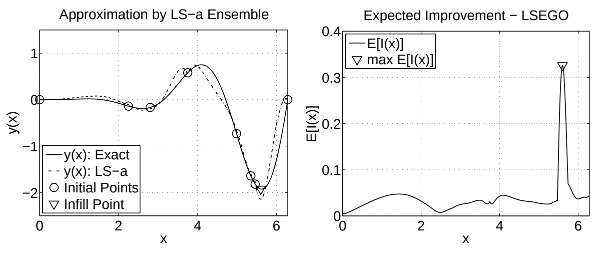
(a) Cycle 01



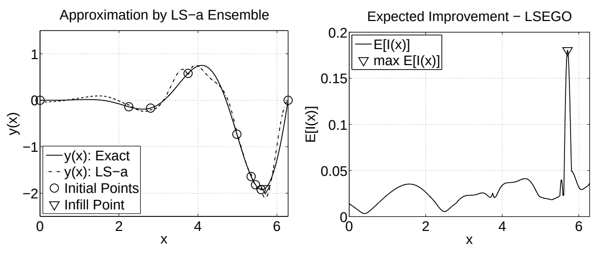
(b) Cycle 02



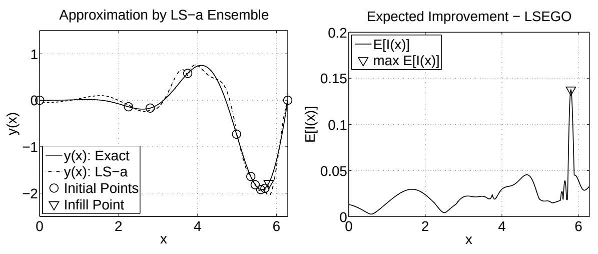
(c) Cycle 03



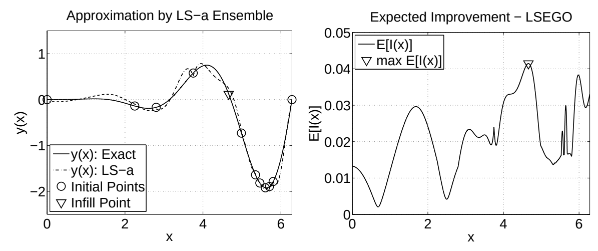
(d) Cycle 04



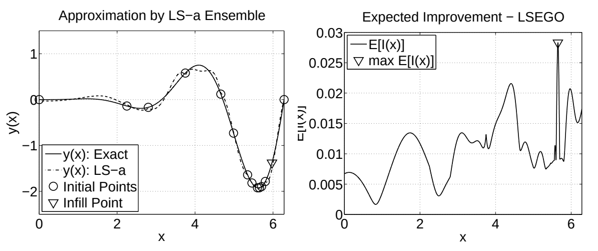
(e) Cycle 05



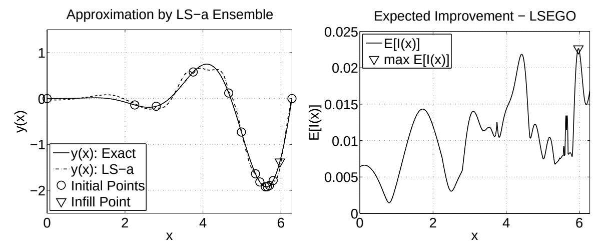
(f) Cycle 06



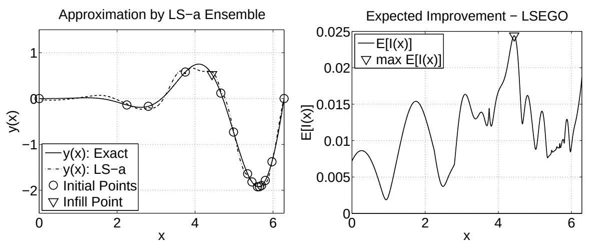
(g) Cycle 07



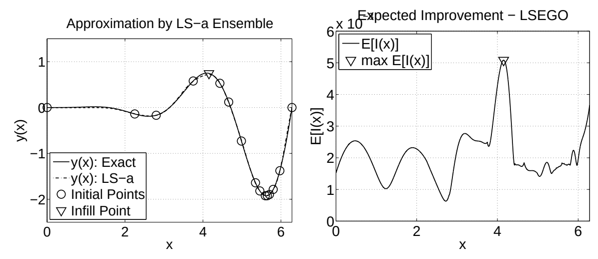
(h) Cycle 08



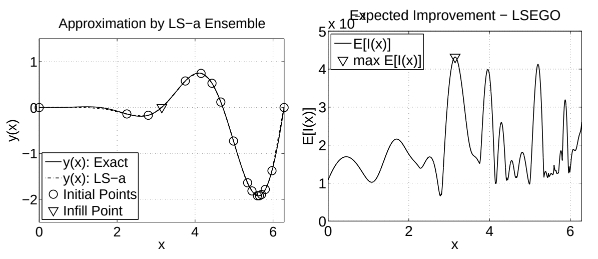
(i) Cycle 09



(j) Cycle 10

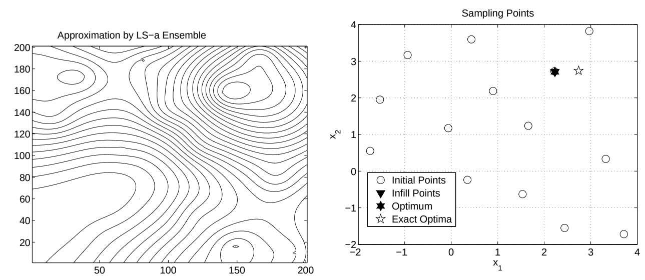


(k) Cycle 11

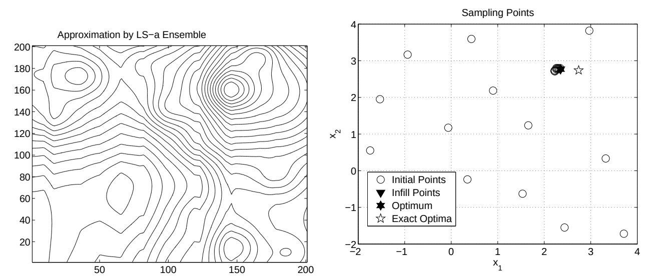


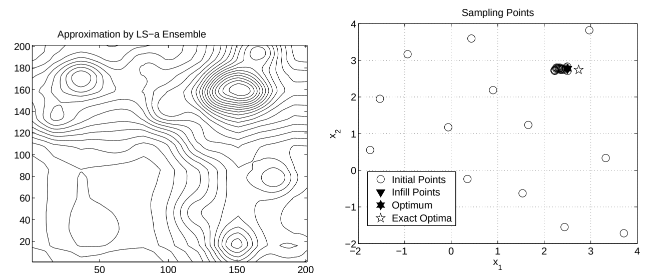
(l) Cycle 12

Fig. 1 Evolution of approximation $y(x)$ vs. $\hat{y}(x)$ and expected improvement $E[I(x)]$ for $y(x) = \frac{1}{700}(-2x+5x^2+7x^3)\sin(2x)$ with $Np = 1$ infill point per optimization cycle with LSEGO algorithm. The initial sampling points are at $\chi = [0, 2.25, 2.8, 3.75, 2\pi]$ and the LS-a ensemble is used with four metamodels: PRS ($ID = 1$), KRG ($ID = 2$), RBNN ($ID = 3$) and SVR ($ID = 4$), see Tab. 1.



(a) Cycle 01


(b) Cycle 10


(c) Cycle 20

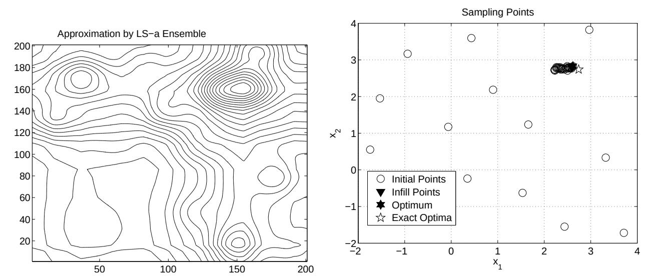
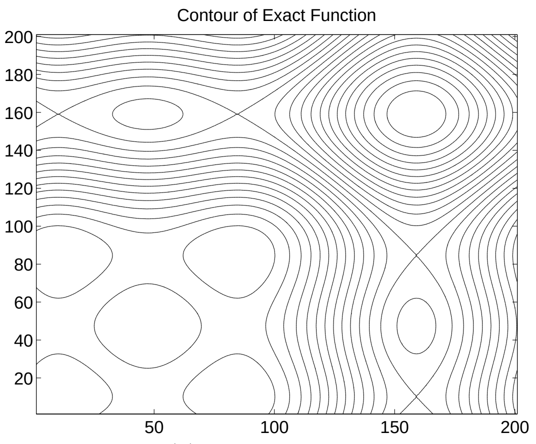
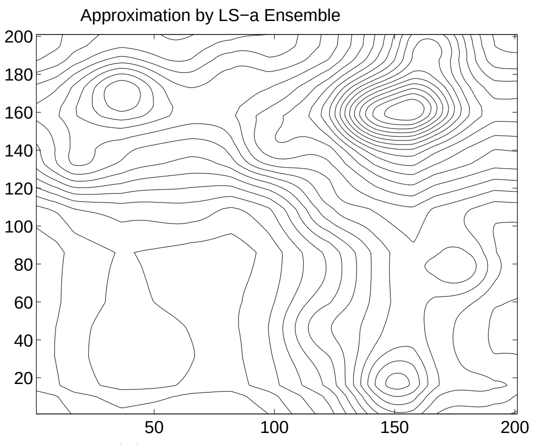

(d) Cycle 30

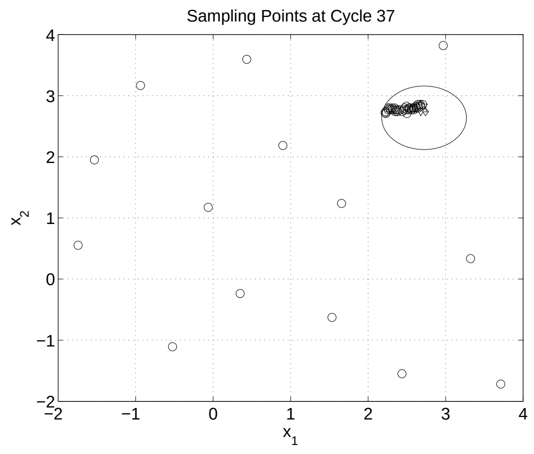
Fig. 2 Evolution of approximation of Giunta-Watson function (2 variables), with LSEGO and $Np = 1$ infill point per optimization cycle. The 15 initial sampling points were generated with Latin Hypercube Matlab function `lhsdesign`, and the LS-a ensemble is used with four metamodels: PRS ($ID = 1$), KRG ($ID = 2$), RBNN ($ID = 3$) and SVR ($ID = 4$), see Tab. 1.



(a) Exact Contour

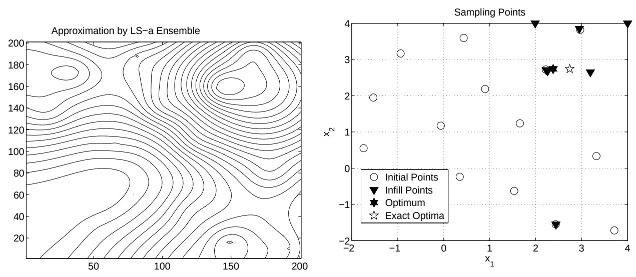


(b) Cycle 37: Approximation

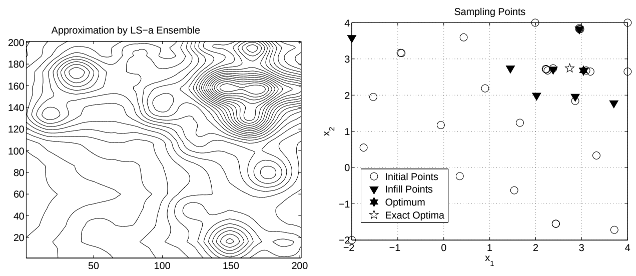


(c) Cycle 37: Sampling Points

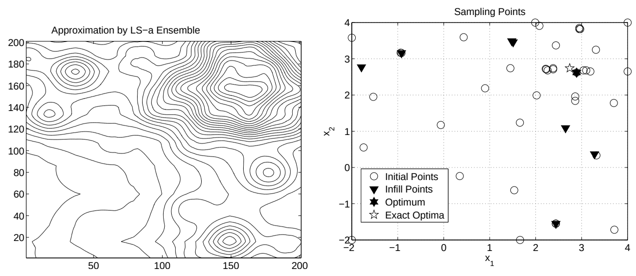
Fig. 3 Detail at Cycle 37 for the evolution of approximation of Giunta-Watson function (2 variables), with LSEGO and $Np = 1$ infill point per optimization cycle.



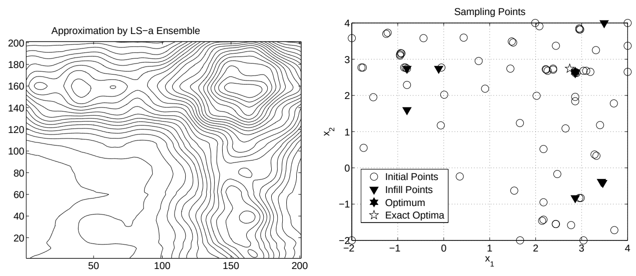
(a) Cycle 01



(b) Cycle 03

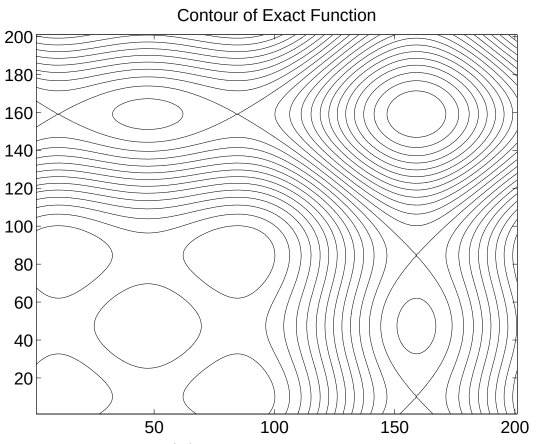


(c) Cycle 05

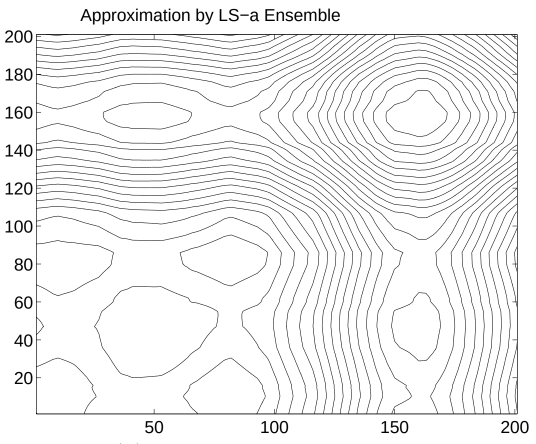


(d) Cycle 10

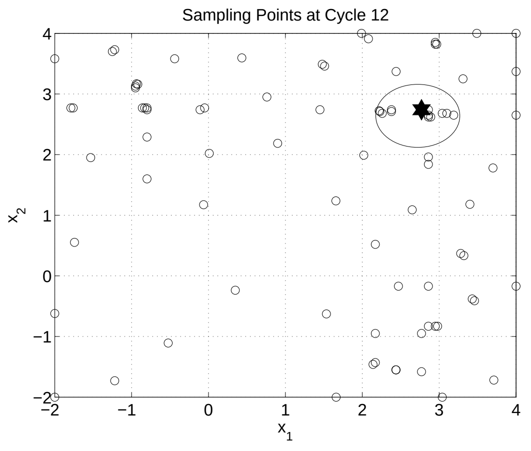
Fig. 4 Evolution of approximation of Giunta-Watson function (2 variables), with LSEGO-8: $Np = 8$ infill points per optimization cycle and the LS-a ensemble is used with four metamodels: PRS ($ID = 1$), KRG ($ID = 2$), RBNN ($ID = 3$) and SVR ($ID = 4$), see Tab. 1.



(a) Exact Contour

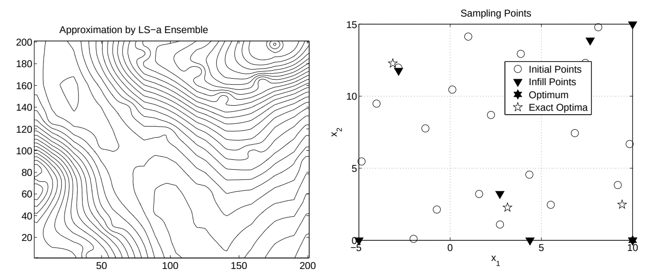


(b) Cycle 12: Approximation

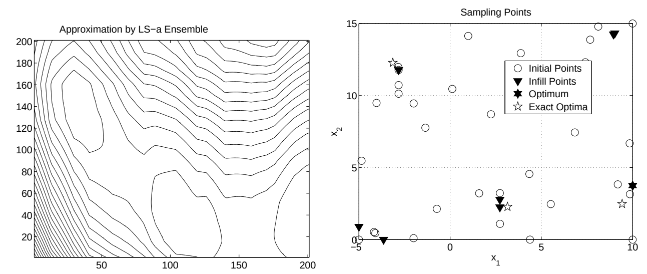


(c) Cycle 12: Sampling Points

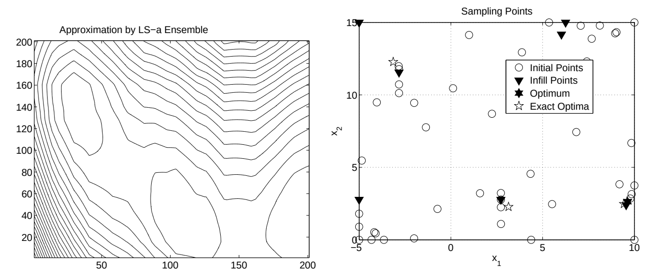
Fig. 5 Detail at Cycle 12 for the evolution of approximation of Giunta-Watson function (2 variables), with LSEGO-8: $Np = 8$ infill points per optimization cycle.



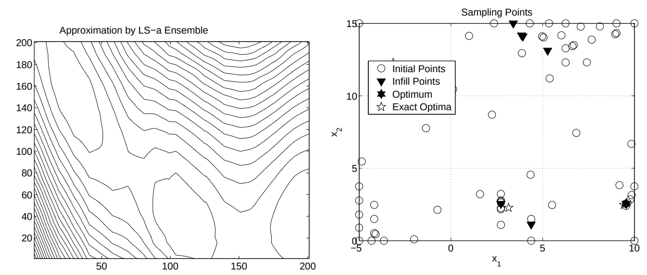
(a) Cycle 01



(b) Cycle 03

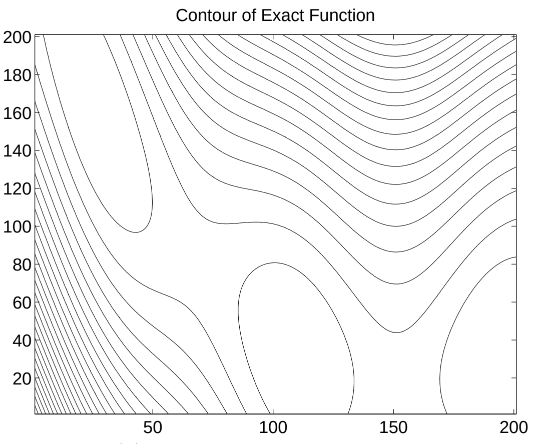


(c) Cycle 05

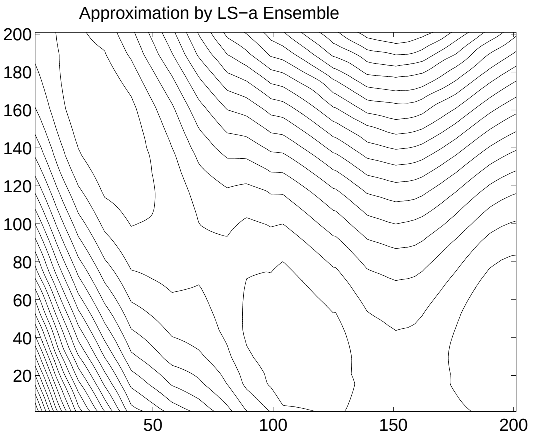


(d) Cycle 10

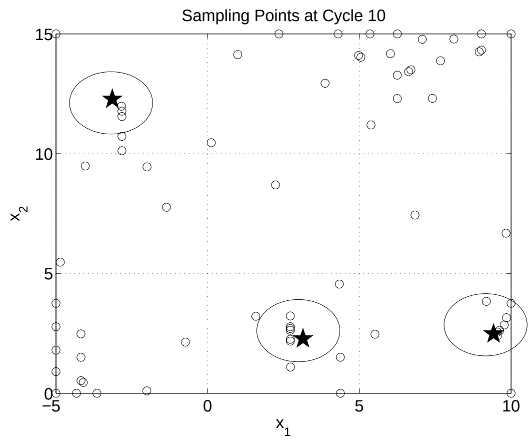
Fig. 6 Evolution of approximation of Branin-Hoo function (2 variables), with LSEGO-8: $Np = 8$ infill points per optimization cycle and the LS-a ensemble is used with four metamodels: PRS ($ID = 1$), KRG ($ID = 2$), RBNN ($ID = 3$) and SVR ($ID = 4$), see Tab. 1.



(a) Cycle 10: Exact Contour



(b) Cycle 10: Approximation



(c) Cycle 10: Sampling Points

Fig. 7 Detail at Cycle 10 for the evolution in the approximation of Branin-Hoo function (2 variables) with LSEGO-8: $Np = 8$ infill points per optimization cycle.

4 Numerical experiments

In this work the main objective is to compare the performance of the proposed algorithm LSEGO versus the traditional EGO. The approach followed for analysis here was based mainly on the work by Viana et al (2013), with specific changes in in the overall setup for the numerical experiments. In addition we applied LSEGO in the optimization constrained optimization of analytical benchmark functions, with the approach outlined in Section 2.4, based on Eq. (6).

4.1 Computer implementation

We used the Matlab[3] SURROGATES Toolbox v2.0 (ref. Viana (2009)), as platform for implementation and tests. See Appendix C for details.

In our previous work, Ferreira and Serpa (2015), we implemented routines for LS ensemble of metamodels. In the present work we extended the implementations to include the standard EGO and LSEGO algorithms as well, as described respectively in Section 2.3 and Section 3.5.

The numerical implementation has been performed with Matlab v2009, on a computer Intel(R) Core(TM) i7-3610QM, CPU 2.30GHz, 8Gb RAM, 64bits, and operational system Windows 7.

4.2 Experimental Setup

4.2.1 Analytical benchmark functions

We used three well known analytical functions with different number of variables (n_v) for testing the optimization algorithms: Branin-Hoo ($n_v = 2$), Hartman-3 ($n_v = 3$) and Hartman-6 ($n_v = 6$). See Appendix B for the respective equations.

For the constrained optimization experiments we generated the constraints by following the approach used in Forrester et al (2008) to test the constrained expected improvement formulation.

That is, for Branin-Hoo function, let $x_1^* = 3\pi$ and $x_2^* = 2.475$ be the coordinates of one of the three local optima, then we write the normalized constraint as follows

$$g(x_1, x_2) = \frac{x_1 x_2}{x_1^* x_2^*} - 1 \geq 0. \tag{10}$$

[3] Matlab a well known and widely used numerical programing platform and it is developed and distributed by The Mathworks Inc., see `www.mathworks.com`.

In this way, by using this hyperbola function, at least one local optimum is forced to lie exactly at the constraint boundary.

The same idea was used for Hartman-3 and Hartman-6 functions, with their respective global optima, listed in Appendix B.

4.2.2 Ensembles of metamodels

The ensemble of metamodels were created with the augmented least squares approach LS-a (ref. Ferreira and Serpa (2015)), with $\eta_{aug} \approx 33\%$ and nine distinct models of type PRS, KRG, RBNN and SVR, by considering the setup presented in Tab. 1. Refer to SURROGATES Toolbox manual (ref. Viana (2009)) for details on the equations and tuning parameters for each of these metamodeling methods.

The EGO algorithm was implemented with the KRG model $ID = 2$ presented on Tab. 1. In case of LSEGO, we used ten permutations of the nine models displayed on Tab. 1 as follows. The first (full) ensemble used all the nine metamodels. For the second partial ensemble we removed the model with $ID = 9$ from the full ensemble. For the third one, the model with $ID = 8$ was removed from the full ensemble and we continue this way up to ten ensembles, i.e., one full LS ensemble ($M = 9$) and nine partial LS ensembles ($\bar{M} = 8$).

4.2.3 Design of experiments

The quality of the approximation by metamodels and the rate of convergence to the optimum is strongly dependent on the number and distribution of the initial sampling points defined in the design space (i.e., DOE). In the cases investigated, as a common practice in comparative studies of metamodeling performance, we repeated each experiment with 100 different DOE, in order to average out the influence of random data on the quality of fit. The detailed setup for the initial DOE for each test problem is presented in Tab. 2. We used the same number of initial points N_{tr} in the DOE as used in Jones et al (1998).

The 100 different initial DOE with N points ($N = N_{tr} + N_{add}$) were created by using the Latin Hypercube Matlab function `lhsdesign`, optimized with `maxmin` criterion set to 1000 iterations. At each cycle of LSEGO, the augmenting points N_{add} are chosen randomly for the full sampling set (N) to generate the LS-a ensemble with constant rate $\eta_{aug} \approx 33\%$.

4.2.4 Setup for EGO algorithms

In the optimization of each test problem (i.e., Branin-Hoo, Hartman-3 and Hartman-6), we repeated EGO

Table 1 Basic metamodels setup for creating the ensembles.

ID	Type	Details
1	PRS	Full quadratic model
2	KRG	Quadratic regression, exponential correlation, $\theta_0 = 10$ and $10^{-2} \leq \theta_i \leq 200$
3	RBNN	$Goal = (0.05\bar{y})^2$, $Spread = 2/5$ and $MN = N$
4	SVR	$C = 100max(\lvert\bar{y} + 3\sigma_y\rvert, \lvert\bar{y} - 3\sigma_y\rvert)$ and $\epsilon = \sigma_y/\sqrt{N}$
5	PRS	Linear model
6	KRG	Linear regression, Gaussian correlation, $\theta_0 = 10$ and $10^{-2} \leq \theta_i \leq 200$
7	RBNN	$Goal = (0.05\bar{y})^2$, $Spread = 1/3$ and and $MN = N/2$
8	SVR	$C = 100max(\lvert\bar{y} + 3\sigma_y\rvert, \lvert\bar{y} - 3\sigma_y\rvert)$, $\epsilon = \sigma_y/\sqrt{N}$, $KernelOptions = 1/2$ and $Loss = Quadratic$
9	KRG	Constant regression, Gaussian correlation, $\theta_0 = 10$ and $10^{-2} \leq \theta_i \leq 200$

Obs.1: KRG: kriging; PRS: polynomial response surface; RBNN: radial basis neural network; SVR: support vector regression.
Obs.2: All other parameters not mentioned are kept with default values.
Obs.3: $\bar{y}$, σ_y and N are respectively: mean and standard deviation of y and number of sampling points.
Obs.4: No attempt has been made in order to fine tuning the surrogates shape parameters.

Table 2 Basic specifications for the initial DOE. n_v: number of variables, N_{tr}: training points and N_{add}: augmenting points for LS-a ensemble.

Test Problem	n_v	N_{tr}	N_{add}
Branin-Hoo	2	21	10
Hartman-3	3	33	16
Hartman-6	6	65	32

and LSEGO $N_{rep} = 100$ times, in order to average out the effect of different initial DOE on the convergence.

In case of EGO we used the standard case ($N_p = 1$) infill point per cycle. In case of LSEGO we used for Branin-Hoo ($N_p = 2$, 5 and 10) points per cycle and for Hartman functions ($N_p = 10$). The acronym LSEGO varies as function of N_p, e.g., LSEGO-1 stands for LSEGO with $N_p = 1$ and LSEGO-10 stands for LSEGO with $N_p = 10$.

In order to compare the rate of convergence (y_{min} vs. cycles), the total number of cycles allowed to run was set 5 for Branin-Hoo, 10 for Hartman-3 and 15 for Hartman-6. For comparison of variability of y_{min} at each cycle, as function of different DOE, we used *boxplots*.[4]

In order to compare the *level of improvement* (L_{imp}) versus number of function evaluations (f_{eval}), for all test problems, it was considered $f_{eval}^{max} = 50$, with $N_{rep} = 25$ repetitions. In this case L_{imp} is calculated for each cycle k as:

$$L_{imp} = 100\% \times \left| \frac{y_{min}^k - y_{min}^0}{y_{min}^{exact} - y_{min}^0} \right|, \quad (11)$$

[4] Boxplot is a common statistical graph used for visual comparison of the distribution of different variables in a same plane. The box is defined by lines at the lower quartile (25%), median (50%) and upper quartile (75%) of the data. Lines extending above and upper each box (*whiskers*) indicate the spread for the rest of the data out of the quartiles definition. If existent, outliers are represented by plus signs "+", above/below the whiskers. We used the Matlab function `boxplot` (with default parameters) to create the plots.

where y_{min}^0 is the minimum for the initial sample, y_{min}^k is the minimum value at cycle k and y_{min}^{exact} is the exact minimum value.

4.2.5 Maximization of the expected improvement functions

In case of standard EGO (with one point per cycle) we used the Matlab built-in genetic algorithm optimizer `ga` with both `PopulationSize` and `Generations` set 100. The `InitialPopulation` was set with size $10n_v$ individuals, chosen by using the function `lhsdesign`, optimized with `maxmin` criterion set to 1000 iterations.

In case of LSEGO with multiple infill points there are many expected improvement functions to maximize per cycle. The use of a genetic algorithm can be quite time consuming in this case. Based on preliminary numerical experiments, we found a good balance in accuracy and computation time by using the Matlab pattern search algorithm `patternsearch`, with $10n_v$ initial points (`X0`), chosen by using the function `lhsdesign`, optimized with `maxmin` criterion set to 1000 iterations.

As discussed in Jones (2001) the EGO-type algorithms tends to generate infill points quite close to each other in several cycles. Sampling points too close can degenerate the approximation of many metamodels, in special KRG and RBF. In addition, infill points too close have low contribution to the exploration of the design space, they can be a waste of resources and in addition lead to a slower convergence for the optimization. We verified this fact in several preliminary numerical experiments (ref. for instance the illustration example of Fig. 2).

In this case, in order to avoid approximation issues during EGO and LSEGO cycles, we generated exceeding infill points per cycle and selected *distinct* N_p with highest expected improvement value. In our tests we found a good balance by generating $2N_p$ candidate infill points and then we used a *clustering* procedure to remove points too close, or even equal to each other. After the clustering selection, the N_p distinct points are added to the sampling space for the next optimization cycle.

In our first tests we used the Matlab function `unique` to remove equal points from the sampling space, but this procedure has shown to be not effective. Then we implemented a clustering selection procedure by using the Matlab function `cluster` with the `distance` criterion. The `cutoff` was based on the maximum distance (d_{max}) among points/clusters in the whole sampling space, by using the Matlab `linkage` function. In preliminary numerical tests we found that `cutoff` around to 10% of d_{max} is effective to remove too close points.

5 Results and discussion

5.1 Comparative study: LSEGO versus EGO

The objective in this test set is to compare the performance of the proposed algorithm LSEGO versus the traditional EGO.

In Fig. 8 it is presented the optimization results (median over 100 runs starting with different experimental designs) for the classical efficient global optimization algorithm EGO (only with kriging and one infill point per cycle) and LSEGO (with LS ensemble of surrogates/metamodels and different number of points per cycle, N_p).

For the three functions studied here, it was observed significant improvement on the reduction of number of cycles for convergence by adding more infill points per cycle. The higher the number of points per cycle, faster the convergence. In addition this effect of accelerated convergence per cycle is more evident as the number of variables increases.

In case of Branin-Hoo function, we tested LSEGO for $N_p = 2$, 5 and 10. For Hartman-3 and Hartman-6 it is presented only for $N_p = 10$. In case of Hartman functions we found a similar trend, as presented in Viana et al (2013), with little difference in convergence for $N_p = 5$ and $N_p = 10$. In this way we removed $N_p < 10$ from the scope for these cases.

Fig. 9 is the counterpart of Fig. 8, by plotting the results versus function evaluations instead of optimization cycles. The idea is to compare LSEGO and EGO in terms of computational costs, measured by the number of simulations (function evaluations) required for the same level of improvement on the objective function at each optimization cycle.

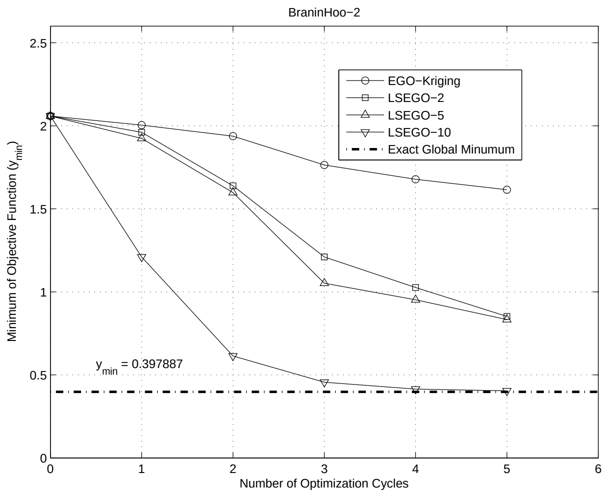
As showed in the curves of Fig. 9, we can observe that for Branin-Hoo and Hartman-3 (low to mid dimensions) at the beginning of the optimization cycles EGO-Kriging has lower cost in terms of number of function evaluations than LSEGO-10 for the same level of improvement, but at some point (e.g., around 70% for Branin-Hoo) the situation is more favorable to LSEGO-10, which is also observed of Hartman-3. In case of the Hartman-6 function, the convergence of EGO-Kriging is quite slow and LSEGO-10 always outperforms EGO-Kriging in terms of function evaluations for the same level of improvement.

These results confirm the fact that by using multiple infill points per cycle with EGO algorithms is, in general, beneficial in terms of delivering results in reasonable processing time (when parallel computation is available), but on the other hand it is not guaranteed that the total computational cost of the optimization process (by the total number of parallel simulations) will be reduced at the end of the day. The trend on the results indicate that this effect is problem dependent on the number of variables and the level of nonlinearity of the problem at hand.

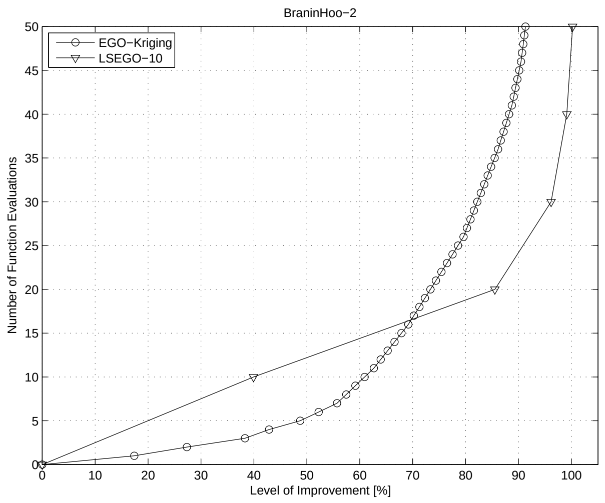
Fig. 10 presents boxplots of the optimization results for 100 experimental designs for LSEGO vs. EGO. Note that at the beginning of the optimization (cycle 1) the minimum solution found has a high dispersion as function on the initial DOE and this dispersion is rapidly decreasing for LSEGO, what is observed in EGO at much lower rate.

Again, in the same direction of the results presented by Viana et al (2013), we found that by using LSEGO with multiples points per cycle a significant reduction in the dispersion of the results (minimum of objective function) can be achieved as long as the optimization cycles evolve. This trend confirms that by using LSEGO with multiple points per cycle reduces the variability and dependence of the optimization results on the initial DOE, what is not assured by using EGO with only one infill point per cycle.

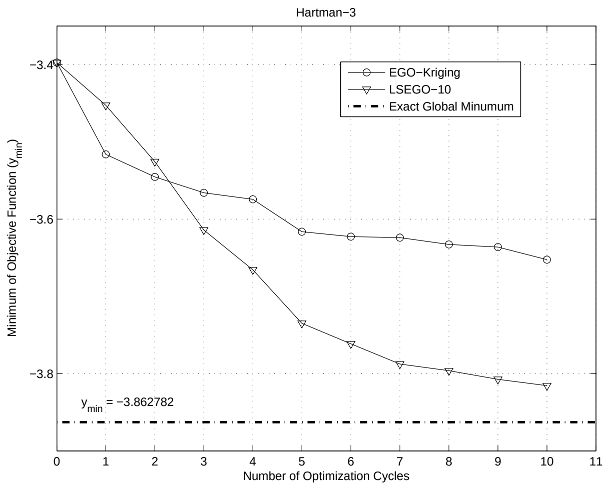
In summary the main objective of the experiments in this set, compiled in Figs. 8, 9 and 10, is to highlight that even by paying more function evaluations per cycle, the behavior of parallel EGO-type algorithms can be more stable with higher chance of convergence and real improvement of the objective function with few optimization cycles/iterations. In other words, as we discussed in the Introduction (Section 1), these results confirm that it can be worthwhile to spend more evaluations in a controlled way in order to improve the overall performance of the optimization process.



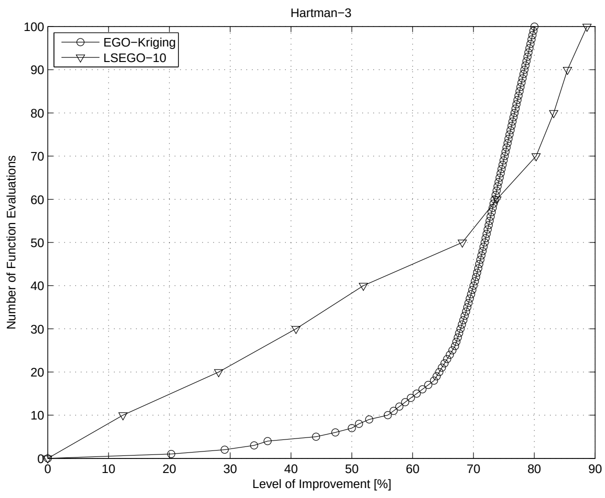
(a) Branin-Hoo, $n_v = 2$



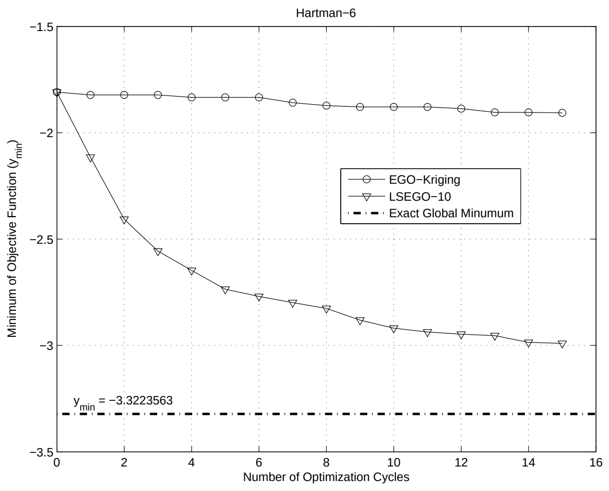
(a) Branin-Hoo, $n_v = 2$



(b) Hartman-3, $n_v = 3$

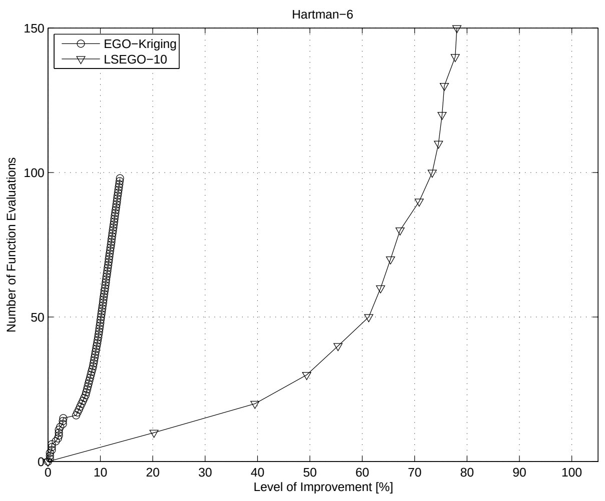


(b) Hartman-3, $n_v = 3$



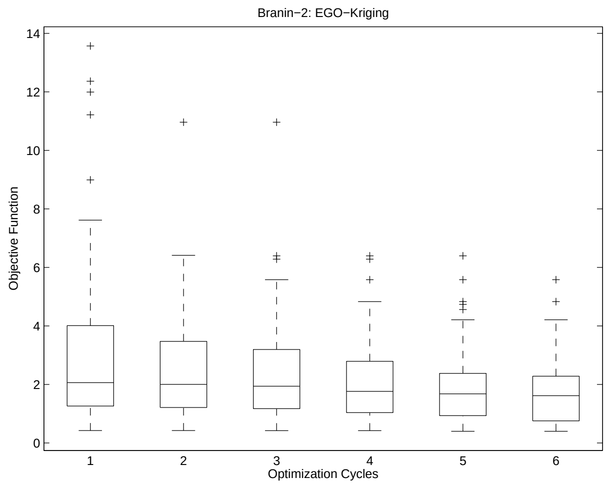
(c) Hartman-6, $n_v = 6$

Fig. 8 Comparison EGO-Kriging versus LSEGO variants for the functions Branin-Hoo, Hartman-3 and Hartman-6. Median (over 100 different initial sampling, DOE) for the efficient global optimization results as function of the number of cycles. The convergence to the exact global minimum is accelerated by adding more points per cycle with LSEGO in all the cases studied.

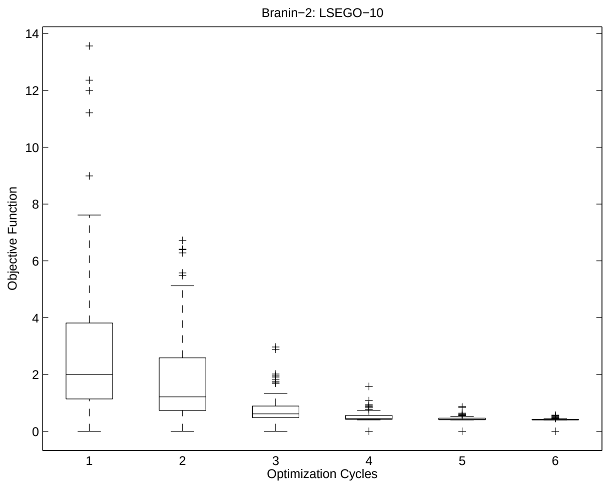


(c) Hartman-6, $n_v = 6$

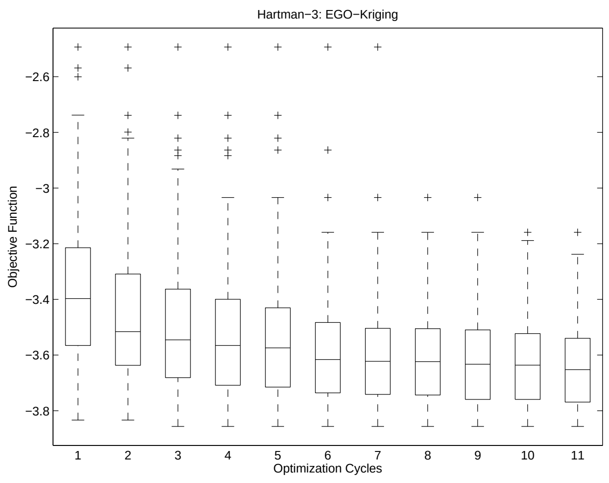
Fig. 9 Comparison EGO-Kriging versus LSEGO-10. Median (over 100 runs with different initIal samples, DOE) for the number of function evaluations versus level of improvement. In case of (Branin-Hoo and Hartman-3, LSEGO required more function evaluations to achieve the same level of improvement at the beginning of the optimization. In case of Hartman-6 LSEGO required less function evaluations for the same improvement in the whole the process.



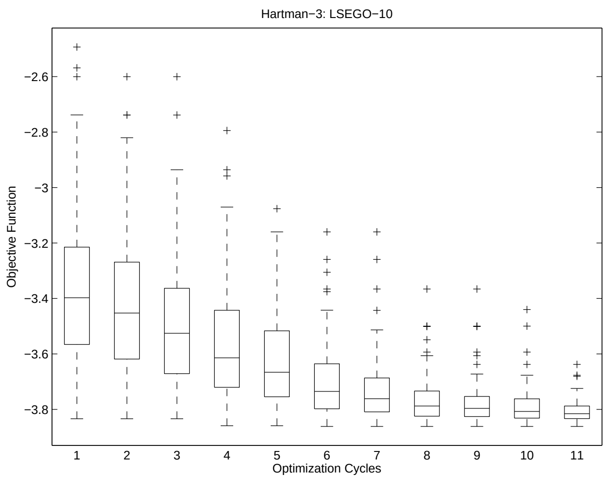
(a) Branin-Hoo, EGO



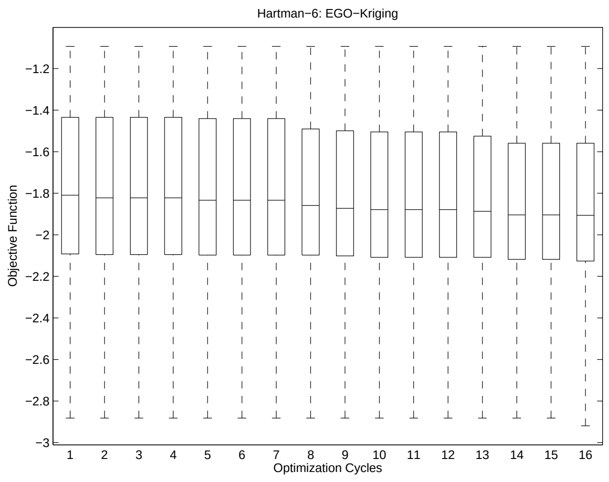
(b) Branin-Hoo, LSEGO-10



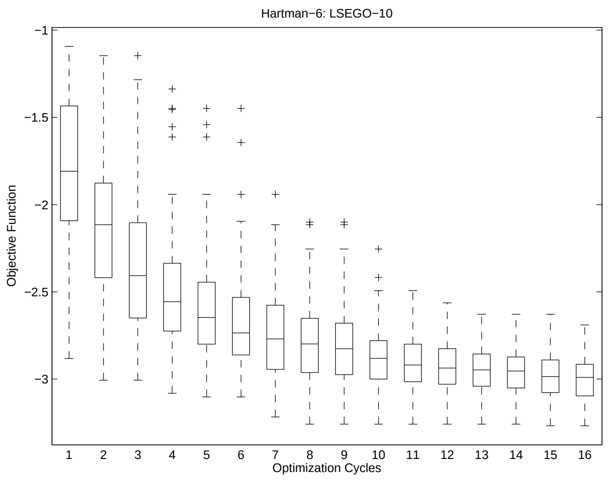
(c) Hartman-3, EGO



(d) Hartman-3, LSEGO-10



(e) Hartman-6, EGO



(f) Hartman-6, LSEGO-10

Fig. 10 Comparison of the convergence of EGO-Kriging (left boxplots) vs. LSEGO-10 (right boxplots), over 100 different initial DOE in each case for the functions Branin-Hoo, Hartman-3 and Hartman-6. The variability of the results is higher as the number of variables increases and the convergence to the optimum is very slow with one infill point and EGO-Kriging. In all the cases, the addition of multiple infill points per optimization cycle with LSEGO-10 accelerates the convergence and also reduce significantly the dispersion of the results as the optimization cycles evolve.

5.2 Results with constrained optimization

5.2.1 Braninh-Hoo

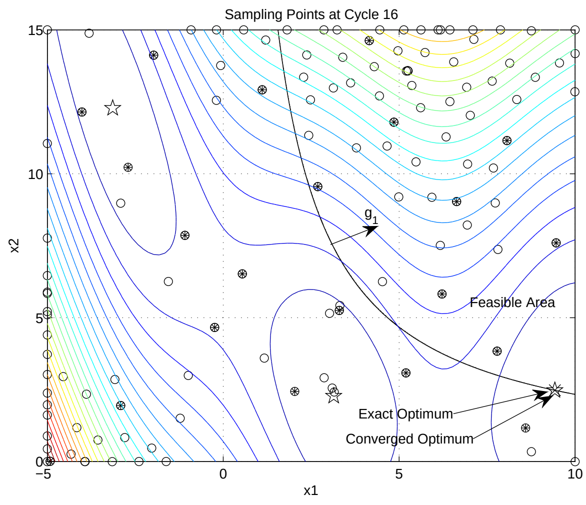


Fig. 11 Optimization results for the constrained Branin-Hoo function. LSEGO-10 converged exactly to the constrained optimum at $\mathbf{x}^* = (9.425, 2.475)$, after 15 cycles. Note the higher density and uniformity of infill points inside than outside the feasible area. The circles "o" are the infill sampling points and the initial DOE points are labeled with asterisks "$*$". The three unconstrained local optima of Branin-Hoo are labeled as "stars".

For the Branhin-Hoo function, the results of the constrained optimization with LSEGO-10 are presented in Fig. 11. In this case, with the constraint defined with Eq. (10), there is only one global optimum in the feasible area, exactly at the constraint boundary at

$$\mathbf{x}^*_{exact} = (9.425, 2.475)\,.$$

Note that the algorithm added infill points at the whole design space, but the density and uniformity of infill points inside is higher than outside the feasible area.

The evolution of the objective function $y(\mathbf{x})$ and the normalized constraint $g(\mathbf{x})$ during the optimization cycles with LSEGO-10 for Branin-Hoo function are presented in Fig. 12. Note that LSEGO-10 started very far from the global minimum $y^* = 16.4266$ and reached fast the neighborhood of $\mathbf{x}^*_{exact}$, at the second optimization cycle, with $y^* = 0.5987$, although no further improvement on $y(\mathbf{x})$ was observed until cycle 14. LSEGO-10 converged exactly to the global optimum at cycle 15.

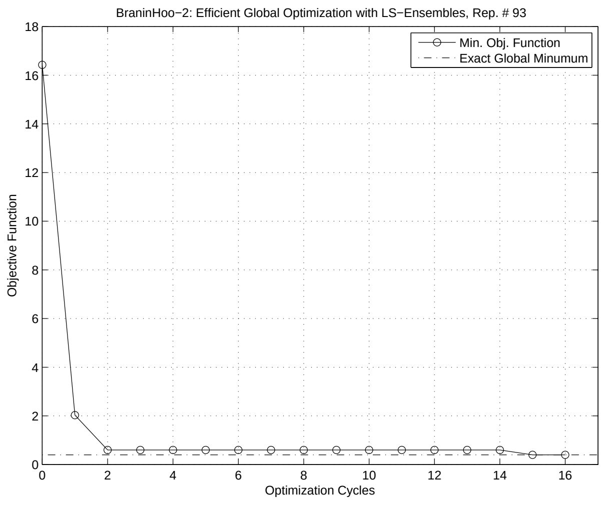
5.2.2 Hartman-3

In the same way, the results for Hartman-3 function is presented in Fig. 13. LSEGO-10 reached the neighborhood of $\mathbf{x}^*_{exact}$ at cycle 5, with $y^* = -3.7899$, i.e., 1.89% error from y^*_{exact}.

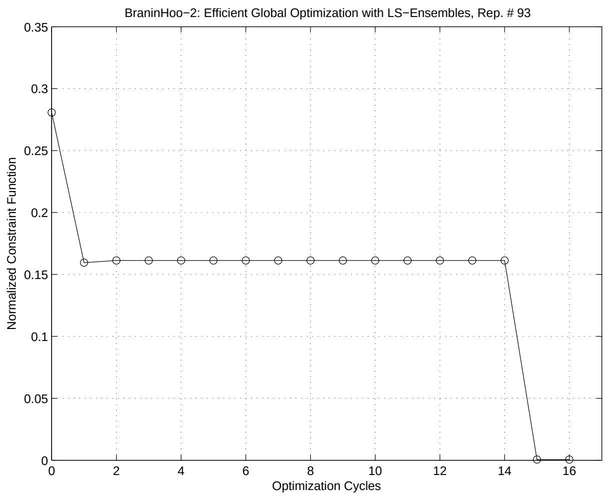
The algorithm evolved in the cycles, by reducing the value of value of $g(\mathbf{x})$ and trying to reach the constraint boundary (i.e., $g(\mathbf{x}) = 0$). At cycle 15, it was found $y^* = -3.8371$, or 0.66% error from y^*_{exact}. At cycle 31, the algorithm reached

$$\mathbf{x}^* = (0.1444, 0.5553, 0.8537)\,,$$

with $y^* = -3.8621$ or 0.02% error from y^*_{exact}, and no further improvement in $y(\mathbf{x})$ or $g(\mathbf{x})$ was found up to 40 cycles.

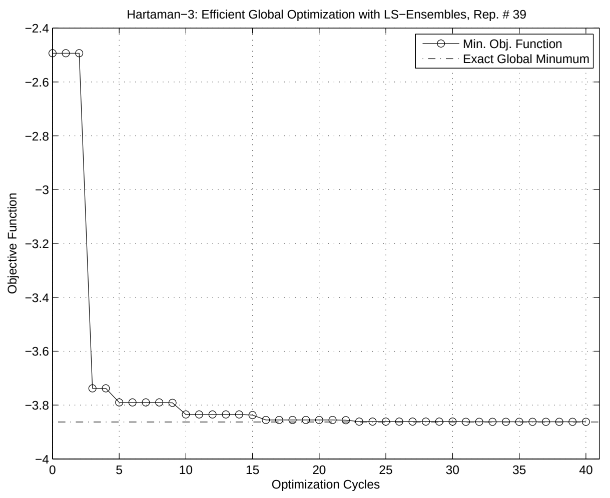


(a) Objective Function $y(\mathbf{x})$

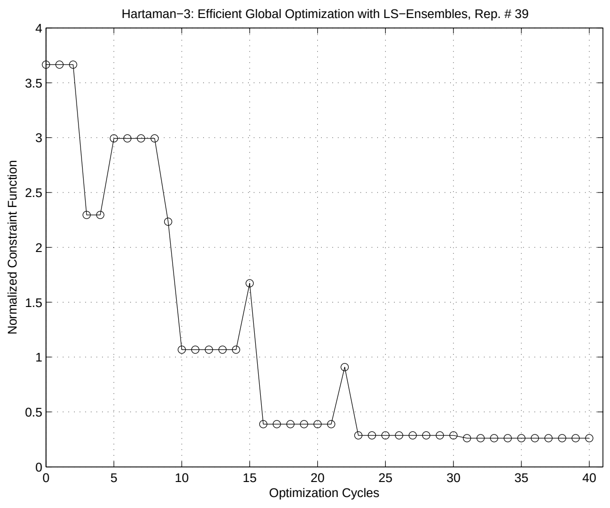


(b) Normalized Constraint $g(\mathbf{x})$

Fig. 12 Evolution objective function $y(\mathbf{x})$ and the normalized constraint $g(\mathbf{x})$ during the optimization cycles with LSEGO-10 for Branin-Hoo function. LSEGO-10 started with $y^* = 16.4266$ and reached $y^* = 0.5987$ at cycle 2. LSEGO-10 converged the global optimum with $y^* = 0.3979$ at cycle 15.

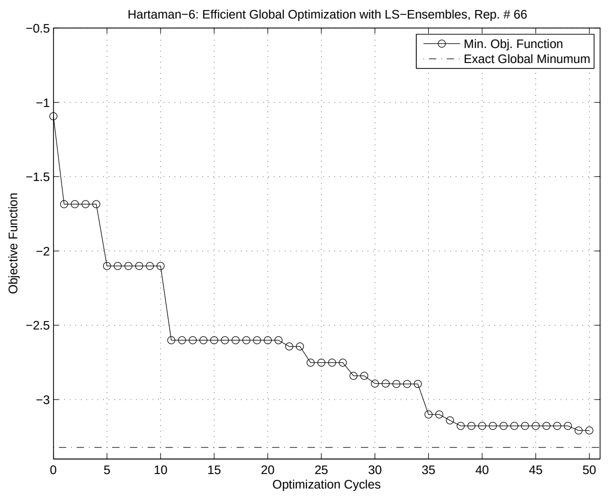


(a) Objective Function $y(\mathbf{x})$

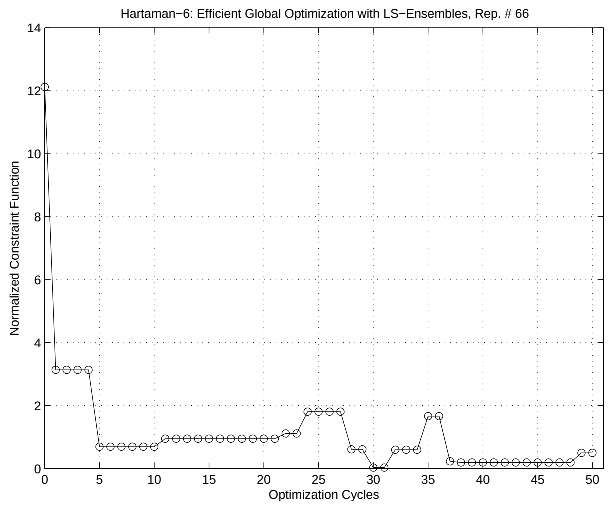


(b) Normalized Constraint $g(\mathbf{x})$

Fig. 13 Evolution of the objective function $y(\mathbf{x})$ and the normalized constraint $g(\mathbf{x})$, during the optimization cycles with LSEGO-10 for Hartman-3 function. LSEGO-10 found $y^* = -3.7899$, i.e., 1.89% error from y^*_{exact} at cycle 5. The algorithm converged at cycle 31 with $y^* = -3.8621$ or 0.02% error from y^*_{exact}.



(a) Objective Function $y(\mathbf{x})$



(b) Normalized Constraint $g(\mathbf{x})$

Fig. 14 Evolution of the objective function $y(\mathbf{x})$ and the normalized constraint $g(\mathbf{x})$, during the optimization cycles with LSEGO-10 for Hartman-6 function. At cycle 35, $y^* = -3.100329$, or 6.68% error from y^*_{exact}. The algorithm was stopped at cycle 50 with little improvement with respect to cycle 38. At this point $y^* = -3.2082$, or 3.44% error from y^*_{exact}.

5.2.3 Hartman-6

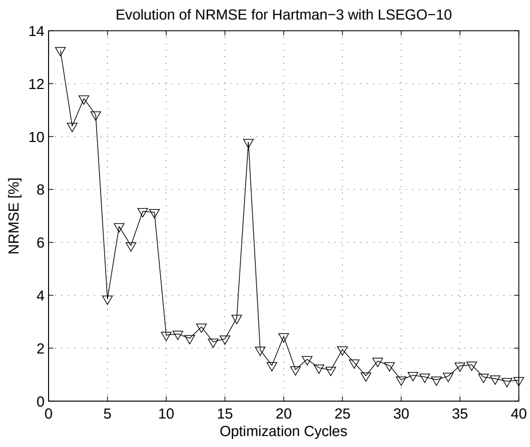
In case of Hartman-6 function, see Fig. 14, the algorithm converged at a slower rate to the neighborhood of $\mathbf{x}^*_{exact}$. At cycle 35, $y^* = -3.100329$, or 6.68% error from y^*_{exact}. At cycle 38, $y^* = -3.177094$, or 4.37% error from y^*_{exact}. We stopped the algorithm at cycle 50 with little improvement with respect to cycle 38, i.e.,

$$\mathbf{x}^* = (0.2688, 0.1547, 0.4508, 0.2890, 0.3433, 0.6541)\,,$$

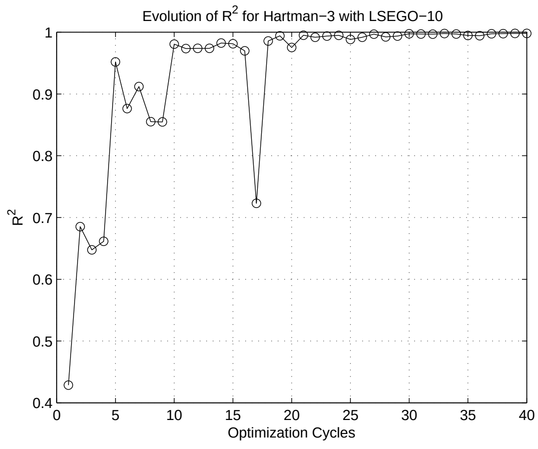
and $y^* = -3.2082$, or 3.44% error from y^*_{exact}.

We repeated these numerical experiments with the analytical benchmark functions and the convergence pattern was nearly the same for different initial sampling points. It is worth noting that in all the cases, the optimization algorithm presented the convergence behavior in steps. Observe this fact in Fig. 12 for Branin-Hoo and with more pronounced effect in Fig. 13 for Hartman-3 and in Fig. 14 for Hartman-6 function.

As discussed in Forrester and Keane (2009), this stepwise behavior can be understood as the algorithm "switching" from the exploitation to exploration modes, during the optimization cycles. In other words, after adding some infill points in the beginning cycles, the quality of fit of the metamodels increase and the algo-



(a) NRMSE, Hartman-3



(b) R^2, Hartman-3

Fig. 15 Evolution of the quality of fit of $y(\mathbf{x})$, by means of NRMSE and R^2, during the optimization cycles with LSEGO-10 for Hartman-3 function. The optimization algorithm switches from exploitation to the exploration mode, during the cycles and the convergence is achieved in steps. The"jumps" downhill in direction to the optimum occur in steps, after the quality of fit of the metamodels of $y(\mathbf{x})$ and $g(\mathbf{x})$ is enough to promote the improvement.

rithm is able to find some improvement in the objective function (exploitation mode).

In the sequence, the algorithm switches to the exploration mode at some cycles, and the next "jump" downhill in direction to the optimum is only achieved after the quality of fit of the metamodels of $y(\mathbf{x})$ and $g(\mathbf{x})$ is enough to promote the next improvement. If we track the quality of fit of the objective function during the optimization cycles, this behavior can be observed.

See for instance the stepwise evolution of the quality of approximation of $y(\mathbf{x})$ by means of NRMSE and R^2, during the optimization cycles for Hartman-3 function in Fig. 15. In this case, at the initial cycles, the quality of fit is erratic at some extent, with jumps at each three or five steps (cycles 3, 5, 10, 15...).

Recall to Fig. 13 and note that these jumps occur simultaneously as the ones at it is observed the main improvements in objective function and for the constraint. When the accuracy of the metamodel reaches stable levels (i.e., $R^2 > 0.9$ and $NRMSE < 2\%$, around cycle 20, the algorithm is quite close to the global optimum and it converges at cycle 25, when the quality of fit is very good (i.e., $R^2 \approx 1$).

In this sense, based on this observed behavior for the algorithm, it is recommended in practical applications to monitor the quality of fit for the metamodels, in parallel to the evolution of the objective and constraint functions, in order to avoid premature or false convergence at suboptimal points. In practical situations, this balance between quality of fit, improvement of objective function and constraints versus total number of sampling points, must be observed for each problem.

In addition, in several practical problems, improvements in the objective or constraints in the order of 10% to 25% are very hard to meet. In such situations, finding one or a set of truly improved designs is much more important than finding the "global" optimum for the problem, and the decision on when to stop the optimization cycles must be taken based on these considerations as well.

These results with the analytical benchmarks showed that LSEGO algorithm works successfully to handle constrained optimization problems as well. A deep numerical investigation must be performed, with different functions (number of variables, nonlinearity, multimodality, etc.) and increasing number of constraints, in order to understand in detail the behavior, convergence properties, advantages and limitations of LSEGO algorithm.

6 Concluding remarks

In this work we presented LSEGO, an approach to drive efficient global optimization (EGO), based on LS (least squares) ensemble of metamodels. By means of LS ensemble of metamodels it is possible to estimate the uncertainty of the prediction by using any kind of metamodels (not only kriging) and provide an estimate for the expected improvement function. In this way, LSEGO is an alternative to find multiple infill points at each cycle of EGO and improve both convergence and prediction quality during the whole optimization process.

At first, we demonstrated the performance of the proposed LSEGO approach with one dimensional and two dimensional analytical functions. The algorithm has been tested with increasing number of infill points per optimization cycle. As more infill points are added per cycle, faster is the convergence to the global optimum (exploitation) and also the quality improvement (predictability) of the metamodel in the whole design domain (exploration).

In a second test set, we compared the proposed LSEGO approach with the traditional EGO (with kriging and a single infill point per cycle). For this intent, we used well known analytical benchmark functions to test optimization algorithms, from two to six variables.

For the problems studied, the proposed LSEGO algorithm has shown to be able to find the global optimum with much less number optimization cycles required by the classical EGO approach. This accelerated convergence was specially observed as the number of variables increased, when the standard EGO can be quite slow to reach the global optimum.

The results also showed that, by using multiple infill points per optimization cycle, driven by LSEGO, the confidence of metamodels prediction in the whole optimization process is improved. It was observed in the boxplots for all cases investigated a variability reduction with respect to the initial sampling space (initial DOE), as more infill points are added during the optimization cycles.

We also compared LSEGO versus standard EGO in terms of the number of function evaluations, which translates directly to computational cost (i.e., number of simulations required). We can observe that for Branin-Hoo and Hartman-3 (low to mid dimensions) at the beginning of the optimization cycles EGO-Kriging has lower cost in terms of number of function evaluations than LSEGO for the same level of improvement, but as the optimization cycle evolve the situation was more favorable to LSEGO, which is also observed of Hartman-3. In case of the Hartman-6 function, the convergence of EGO-Kriging was quite slow and LSEGO outperformed EGO-Kriging in terms of function evaluations for the same level of improvement in the whole optimization process.

In addition we observed that the LSEGO algorithm works successfully to handle constrained optimization problems, in feasible number of optimization cycles. In these constrained problems investigated, LSEGO presented a stepwise convergence pattern, which is common to EGO-type algorithms. In this sense, it is recommended in practical applications to monitor the quality of fit for the metamodels, in parallel to the evolution of the objective and constraint functions, in order to avoid premature or false convergence at suboptimal points.

Although we understand that these are promising and competitive results, a deep numerical investigation must be performed, with different functions (number of variables, nonlinearity, multimodality, etc.) and increasing number of constraints, in order to understand in detail the behavior, convergence properties, advantages and limitations of LSEGO algorithm.

The results achieved in the present work are in accordance with previous work published in the related research area. In this way, LSEGO approach has shown to be a feasible alternative to drive efficient global optimization by using multiple or ensemble of metamodels, not restricted to kriging approximation or single infill point per optimization cycles.

In summary we observed that the behavior of parallel EGO-type algorithms can be more stable with higher chance of convergence and real improvement of the objective function with few optimization cycles/iterations. In other words, it can be worthwhile to spend more evaluations in a controlled way and improve the overall performance of the optimization process.

And, last but not least, as we briefly discussed in Section 2.4, it is worth noting that EGO-type algorithms should be extended to treat properly more general constrained, multiobjective and also robust optimization problems. As future research work, we intend to extend the application of LSEGO approach presented here within the context of constrained and multidisciplinary optimization of a broader set of analytical benchmarks and real world engineering applications. We are already working in this front, with good preliminary results, and our intention is to publish them soon.

Acknowledgements The authors would like to thank Dr. F.A.C. Viana for the prompt help with SURROGATES Toolbox and also for the useful comments and discussions about the preliminary results of this work.

W.G. Ferreira would like to thank Ford Motor Company and also the support of his colleagues at MDO group and

Product Development department that helped on the development of this work, which is part of his doctoral research underway at UNICAMP.

Finally, the authors are grateful for the questions and comments from the journal editors and reviewers. Undoubtedly their valuable suggestions helped to improve the clarity and consistency of the present text.

A The kriging metamodel

Kriging model, originally proposed by Krige (1951), is an interpolating metamodel in which the basis functions, as stated in Eq. 1, are of the form

$$\psi^{(i)} = \psi\left(\left\|\mathbf{x}^{(i)} - \mathbf{x}\right\|\right) = \exp\left(-\sum_{j=1}^{k}\theta_j\left|x^{(i)} - x_j\right|^{p_j}\right), \quad (12)$$

with tuning parameters θ_j and p_j normally determined by maximum likelihood estimates.

With the parameters estimated, the final kriging predictor is of the form

$$\hat{f}(\mathbf{x}) = \hat{\mu} + \boldsymbol{\psi}^T\boldsymbol{\Psi}^{-1}(\mathbf{y} - \mathbf{1}\hat{\mu}), \quad (13)$$

where $\mathbf{y} = \left[y^{(1)}\dots y^{(N)}\right]^T$, $\mathbf{1}$ is a vector of ones, $\boldsymbol{\Psi} = \psi^{(r)(s)}$ is the so called $N \times N$ *matrix of correlations* between the sample data, calculated by means of Eq. 12 as

$$\boldsymbol{\Psi} = \psi\left(\left\|\mathbf{x}^{(r)} - \mathbf{x}^{(s)}\right\|\right) \quad (14)$$

and $\hat{\mu}$ is given by

$$\hat{\mu} = \frac{\mathbf{1}^T\boldsymbol{\Psi}^{-1}\mathbf{y}}{\mathbf{1}^T\boldsymbol{\Psi}^{-1}\mathbf{1}}. \quad (15)$$

One of the key benefits of kriging models is the provision of uncertainty estimate for the prediction (mean squared error, MSE) at each point $\mathbf{x}$, given by

$$\hat{s}^2(\mathbf{x}) = \hat{\sigma}^2\left[1 - \boldsymbol{\psi}^T\boldsymbol{\Psi}^{-1}\boldsymbol{\psi} + \frac{1 - \mathbf{1}^T\boldsymbol{\Psi}^{-1}\mathbf{y}}{\mathbf{1}^T\boldsymbol{\Psi}^{-1}\mathbf{1}}\right], \quad (16)$$

with variance estimated by

$$\hat{\sigma}^2 = \frac{(\mathbf{y} - \mathbf{1}\hat{\mu})^T\boldsymbol{\Psi}^{-1}(\mathbf{y} - \mathbf{1}\hat{\mu})}{N}. \quad (17)$$

Refer to Forrester et al (2008) or Fang et al (2006) for further details on metamodel formulation.

B Analytical benchmark functions

These functions were chosen since they are widely used to validate both metamodeling and optimization methods, as for example in and Jones et al (1998) and Viana et al (2013).

Branin-Hoo

$$y(\mathbf{x}) = \left(x_2 + \frac{5.1x_1^2}{4\pi^2} + \frac{5x_1}{\pi} - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos(x_1) + 10, \quad (18)$$

for the region $-5 \le x_1 \le 10$ and $0 \le x_2 \le 15$. There are 3 minima in this region, i.e., $\mathbf{x}^* \approx (-\pi, 12.275), (\pi, 2.275), (3\pi, 2.475)$ with $f(\mathbf{x}^*) = \frac{5}{4\pi}$.

Hartman

$$y(\mathbf{x}) = -\sum_{i=1}^{4}c_i\exp\left[-\sum_{j=1}^{n_v}a_{ij}(x_j - p_{ij})^2\right], \quad (19)$$

where $x_i \in [0,1]^{n_v}$, with constants c_i, a_{ij} and p_{ij} given in Table 3, for the case $n_v = 3$ (Hartman-3); and in Table 4 Table 5, for the case $n_v = 6$ (Hartman-6).

In case of Hartman-3, there are four local minima,

$$\mathbf{x}_{local} \approx (p_{i1}, p_{i2}, p_{i3}),$$

with $f_{local} \approx -c_i$ and the global minimum is located at

$$\mathbf{x}^* \approx (0.114614, 0.555649, 0.852547),$$

with $f(\mathbf{x}^*) \approx -3.862782$.

In case of Hartman-6, there are four local minima,

$$\mathbf{x}_{local} \approx (p_{i1}, p_{i2}, p_{i3}, p_{i4}, p_{i5}, p_{i6}),$$

with $f_{local} \approx -c_i$ and the global minimum is located at

$$\mathbf{x}^* \approx (0.201690, 0.150011, 0.476874, 0.275332, 0.3111652, 0.657301),$$

with $f(\mathbf{x}^*) \approx -3.322368$.

Table 3 Data for Hartman-3 function.

i	c_i	a_{ij}			p_{ij}		
		$j=1$	2	3	$j=1$	2	3
1	1	3	10	30	0.3689	0.117	0.2673
2	1.2	0.1	10	35	0.4699	0.4387	0.747
3	3	3	10	30	0.1091	0.8732	0.5547
4	3.2	0.1	10	35	0.03815	0.5743	0.8828

Table 4 Data for Hartman-6 function, c_i and a_{ij}.

i	c_i	a_{ij}					
		$j=1$	2	3	4	5	6
1	1	10	3	1	3.5	1.7	8
2	1.2	0.05	10	17	0.1	8	14
3	3	3	3.5	1.7	10	17	8
4	3.2	17	8	0.05	10	0.1	14

Table 5 Data for Hartman-6 function, p_{ij}.

i	p_{ij}					
	$j=1$	2	3	4	5	6
1	0.1312	0.1696	0.5569	0.0124	0.8283	0.5886
2	0.2329	0.4135	0.8307	0.3736	0.1004	0.9991
3	0.2348	0.1451	0.3522	0.2883	0.3047	0.665
4	0.4047	0.8828	0.8732	0.5743	0.1091	0.0381

Giunta-Watson This is the "noise-free" version of the function used by Giunta and Watson (1998)

$$y(\mathbf{x}) = \sum_{i=1}^{n_v}\left[\frac{3}{10} + \sin\left(\frac{16}{15}x_i - 1\right) + \sin^2\left(\frac{16}{15}x_i - 1\right)\right], \quad (20)$$

where $\mathbf{x} \in [-2, 4]^{n_v}$.

C SURROGATES Toolbox

The SURROGATES Toolbox (ref. Viana (2009)) is a Matlab based toolbox that aggregates and extends several open-source tools previously developed in the literature for design and analysis of computer experiments, i.e., metamodeling and optimization. We used the version v2.0, but v3.0 already includes EGO variants[5].

The SURROGATES Toolbox uses the following collection of third party software published: SVM by Gunn (1997), DACE by Lophaven et al (2002), GPML by Rasmussen and Williams (2006), RBF by Jekabsons (2009), and SHEPPACK by Thacker et al (2010). The compilation in a single framework has been implemented and applied in previous research by Viana and co-workers, as for example Viana et al (2009) and Viana (2011).

[5] Further details and recent updates of SURROGATES Toolbox refer to the website: `https://sites.google.com/site/srgtstoolbox/`.

References

Fang KT, Li R, Sudjianto A (2006) Design and Modeling for Computer Experiments. Computer Science and Data Analysis Series, Chapman & Hall/CRC, Boca Raton, USA

Ferreira WG, Serpa AL (2015) Ensemble of metamodels: the augmented least squares approach. Structural and Multidisciplinary Optimization 53(5):1019–1046

Forrester A, Keane A (2009) Recent advances in surrogate-based optimization. Progress in Aerospace Sciences 45:50–79

Forrester A, Sóbester A, Keane A (2008) Engineering Desing Via Surrogate Modelling - A Practical Guide. John Wiley & Sons, United Kingdom

Ginsbourger D, Riche RL, Carraro L (2010) Kriging is well-suited to parallelize optimization. In: Computational Intelligence in Expensive Optimization Problems - Adaptation Learning and Optimization, Springer, vol 2, pp 131–162

Giunta AA, Watson LT (1998) Comparison of approximation modeling techniques: polynomial versus interpolating models. In: 7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, AIAA-98-4758, pp 392–404

Gunn SR (1997) Support vector machines for classification and regression. Technical Report. Image, Speech and Inteligent Systems Research Group. University of Southhampton, UK

Han ZH, Zhang KS (2012) Surrogate-Based Optimization - Real-World Application of Genetic Algorithms, ISBN 978-953-51-0146-8 edn. InTech, Dr. Olympia Roeva - Editor, Shanghai, China

Henkenjohann N, Kukert J (2007) An efficient sequential optimization approach based on the multivariate expected improvement criterion. Quality Engineering 19(4):267–280

Jekabsons G (2009) RBF: Radial basis function interpolation for matlab/octave. Riga Technical University, Latvia, version 1.1 ed.

Jones DR (2001) A taxonomy of global optimization methods based on response surfaces. Journal of Global Optimization 21:345–383

Jones DR, Schonlau M, Welch WJ (1998) Efficient global optimization of expensive black-box functions. Journal of Global Optimization 13:455–492

Jurecka F (2007) Optimization based on metamodeling techniques. PhD thesis, Technische Universität München, München-Germany

Koziel S, Leifesson L (2013) Surrogate-Based Modeling and Optimization - Applications in Engineering. Springer, New York, USA

Krige DG (1951) A statistical approach to some mine valuations and allied problems at the witwatersrand. Master's thesis, University of Witwatersrand, Witwatersrand

Lophaven SN, Nielsen HB, Sondergaard J (2002) DACE - a matlab kriging toolbox. Tech. Rep. IMM-TR-2002-12, Technical University of Denmark

Mockus J (1994) Application of bayesian approach to numerical methods of global and stochastic optimization. Journal of Global Optimization 4:347–365

Ponweiser W, Wagner T, Vincze M (2008) Clustered multiple generalized expected improvement: A novel infill sampling criterion for surrogate models. In: Wang J (ed) 2008 IEEE World Congress on Computational Intelligence, IEEE Computational Intelligence Society, IEEE Press, Hong Kong, pp 3514–3521

Queipo NV, et al (2005) Surrogate-based analysis and optimization. Progress in Aerospace Sciences 41:1–28

Rasmussen CE, Williams CK (2006) Gaussian Processes for Machine Learning. The MIT Press

Schonlau M (1997) Computer experiments and global optimization. PhD thesis, University of Waterloo, Watterloo, Ontario, Canada

Simpson TW, Toropov V, Balabanov V, Viana FAC (2008) Design and analysis of computer experiments in multidisciplinary design optimization: A review of how far we have come - or not. In: 12th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Victoria, British Columbia

Sóbester A, Leary SJ, Keane A (2004) A parallel updating scheme for approximating and optimizing high fidelity computer simulations. Structural and Multidisciplinary Optimization 27:371–383

Thacker WI, Zhang J, Watson LT, Birch JB, Iyer MA, Berry MW (2010) Algorithm 905: SHEPPACK: modified shepard algorithm for interpolation of scattered multivariate data. ACM Transactions on Mathematical Software 37(3):1–20

Venkatararaman S, Haftka RT (2004) Structural optimization complexity: what has moore's law done for us? Structural and Multidisciplinary Optimization 28:375–287

Viana FAC (2009) SURROGATES toolbox user's guide version 2.0 (release 3). Available at website: `http://fchegury.googlepages.com`

Viana FAC (2011) Multiples surrogates for prediction and optimization. PhD thesis, University of Florida, Gainesville, FL, USA

Viana FAC, Haftka RT, Steffen V (2009) Multiple surrogates: how cross-validation error can help us to obtain the best predictor. Structural and Multidisciplinary Optimization 39(4):439–457

Viana FAC, Haftka RT, Watson LT (2013) Efficient global optimization algorithm assisted by multiple surrogates techniques. Journal of Global Optimization 56:669–689