METHODOLOGY FOR GLOBAL OPTIMIZATION OF COMPUTATIONALLY EXPENSIVE DESIGN PROBLEMS

A Thesis
Presented to
The Academic Faculty

by

Stefanos Koullias

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Aerospace Engineering

Georgia Institute of Technology
August 2013

METHODOLOGY FOR GLOBAL OPTIMIZATION OF COMPUTATIONALLY EXPENSIVE DESIGN PROBLEMS

Approved by:

Dr. Dimitri N. Mavris, Advisor
School of Aerospace Engineering
Georgia Institute of Technology

Dr. Daniel P. Schrage
School of Aerospace Engineering
Georgia Institute of Technology

Dr. Brian J. German
School of Aerospace Engineering
Georgia Institute of Technology

Dr. Kelly Griendling
School of Aerospace Engineering
Georgia Institute of Technology

Dr. Sankaran Mahadevan
Department of Civil and
Environmental Engineering
Vanderbilt University

Date Approved: May 31 2013

ACKNOWLEDGEMENTS

The author would like to express his sincere appreciation to his advisor, Dr. Dimitri N. Mavris, for his guidance and academic and financial support to pursue this research. Appreciation and thanks are extended to all members of the academic committee, including Dr. Dimitri N. Mavris, Dr. Daniel P. Schrage, Dr. Brian J. German, Dr. Kelly Griendling, and Dr. Sankaran Mahadevan for their careful review of this manuscript and suggestions to improve the quality of this research. The approval of the committee is greatly valued by the author.

The author would like to express his gratitude to Dr. Santiago Balestrini Robinson for his efforts to obtain research funds and for his careful review and co-authorship of several published journal articles and conference papers. The general efforts of Dr. Santiago Balestrini Robinson have improved the research abilities of the author.

Finally, the author would like to thank Ms. Kelly Cooper and the Office of Naval Research for sponsoring the author's research from 2007 – 2012, and for Ms. Kelly Cooper's guidance through the many stages of the sponsored research.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS . . . . **iii**

LIST OF TABLES . . . . **viii**

LIST OF FIGURES . . . . **x**

LIST OF SYMBOLS AND ABBREVIATIONS . . . . **xiv**

SUMMARY . . . . **xix**

I MOTIVATION **1**

1.1 Design Of Advanced Aircraft Concepts 1

1.2 Technical Challenges 5

1.3 Research Focus & Approach 9

II FUNCTION APPROXIMATION TECHNIQUES **15**

2.1 Selection Of Surrogate Model 15

2.2 Gaussian Process Models 19

2.2.1 Mathematical Background 19

2.2.2 Training The GP Model 24

2.2.3 Limitations & Scalability 26

2.2.4 Inadequacy Of Likelihood-Based Approaches 27

2.2.5 Fully Bayesian Approach 29

2.2.6 Contribution: Method For Assigning Hyperparameter Priors 35

2.3 Radial Basis Functions 46

2.4 Kernel Machines 47

III INFILL SAMPLING CRITERIA **50**

3.1 Overview Of Existing Algorithms 50

3.2 Improvement-Based Infill Sampling Criteria 52

3.2.1 Probability Of Improvement 52

3.2.2 Expected Improvement 57

3.2.3 Fully Bayesian Expected Improvement 62
3.3 One-Stage Methods 63
3.3.1 One-Stage EGO 63
3.3.2 Method Of Gutmann 66
3.4 Other Methods 73
3.4.1 CORS 73
3.4.2 DIRECT 74
IV METHODS FOR HANDLING NONLINEAR CONSTRAINTS 78
4.1 Penalty Methods 78
4.2 Bayesian Methods 80
4.2.1 Method Of Schonlau et al. 80
4.2.2 Expected Violation 83
4.3 DIRECT 83
4.4 Other Methods 86
V ADDITIONAL CONSIDERATIONS 88
5.1 Experimental Designs 88
5.2 Initially Infeasible Designs 89
5.3 Missing Data 90
5.4 Sampling In The Presence Of Noise 92
5.5 Stopping Rules 96
VI METHODOLOGY FOR GLOBAL OPTIMIZATION OF COMPUTATIONALLY EXPENSIVE DESIGN PROBLEMS 99
6.1 fBcEGO: Fully Bayesian Constrained Efficient Global Optimization . 99
6.2 fBcEGO Specification 101
6.3 Methodology 102
6.3.1 Step 1: Formal Problem Statement 103
6.3.2 Step 2: Select Initial Design 104
6.3.3 Step 3: Select Stopping Criteria 104

6.3.4 Step 4: Select Covariance Function 104

6.3.5 Step 5: Execute fBcEGO 104

6.3.6 Step 6: Visualization & Analysis 105

VII HYPOTHESIS & TESTING PLAN 108

7.1 Research Questions & Hypotheses 108

7.2 Testing Plan 111

7.3 Test Problems 112

7.4 Competing Algorithms 123

7.5 Metrics Of Performance 126

7.6 Experiments & Test Matrices 129

VIII RESULTS 135

8.1 Experimental Setup & Implementation Details 135

8.1.1 Experimental Designs 135

8.1.2 Model Fitting 135

8.1.3 Optimization Of Infill Sampling Criteria 136

8.1.4 Target Values 137

8.1.5 Stopping Criteria 137

8.1.6 Design Space Symmetry 138

8.2 Experiment 1: Influence Of Initial Design On Performance 138

8.3 Experiment 2: Performance Of ISC On Bound Constrained Problems 141

8.4 Experiment 3: Handling Of Nonlinear Constraints 150

8.5 Experiment 4: Noise Corrupted Observations 156

8.6 Experiment 5: Nonsmooth Problems 161

8.7 Experiment 6: Hard Constraints 169

8.8 Additional Findings 169

8.8.1 Effect Of Prior Density On The Performance Of fBcEGO . . . 169

IX APPLICATIONS 172

9.1 Airfoil Section Design 172

9.1.1 Background & Challenges 173

9.1.2 Competing Methods 175

9.1.3 Methodology Applied To Two-Variable Optimization 176

9.1.4 Methodology Applied To Four-Variable Optimization 183

9.1.5 Conclusions 187

9.2 Design Of A Notional 70-Passenger Aircraft 191

9.2.1 Background & Challenges 191

9.2.2 Flight Optimization System (FLOPS) [65] 194

9.2.3 Competing Methods 195

9.2.4 Methodology Applied To Aircraft Design Problem 195

9.2.5 Conclusions 203

X CONCLUSIONS 211

10.1 Summary Of Contributions 211

10.2 Revisiting The Research Questions & Hypotheses 212

10.3 Recommendations 216

10.4 Suggestions For Further Research 218

APPENDIX A — TEST PROBLEMS 219

APPENDIX B — BAYESIAN MONTE CARLO 241

REFERENCES 247

LIST OF TABLES

1 Computationally expensive black-box optimization problems from the literature . . . 8

2 Matrix of alternatives for the technical challenges . . . 11

3 Relative capabilities of surrogate models . . . 17

4 Overview of existing algorithms . . . 52

5 Experimental designs . . . 90

6 Matrix of alternatives for algorithm components . . . 100

7 Test problem classification scheme . . . 116

8 Condensed list of bound constrained global optimization test problems 117

9 Condensed list of nonlinearly constrained global optimization test problems . . . 117

10 Condensed list of nonsmooth test problems . . . 118

11 Condensed list of noise corrupted test problems . . . 118

12 Competing algorithms for bound constrained test problems . . . 124

13 Competing algorithms for nonlinearly constrained test problems . . . 125

14 Competing GP algorithms for noise corrupted test problems . . . 125

15 Competing algorithms for nonsmooth test problems . . . 125

16 Test matrix for Experiment 6 . . . 134

17 Successful tests as a function of initial design . . . 140

18 Low dimensional design problems used for tests over different initial designs . . . 147

19 Number of function evaluations to achieve an accuracy of 1%; low dimensional problems . . . 149

20 Number of function evaluations to achieve an accuracy of 0.1%; low dimensional problems . . . 149

21 Number of function evaluations required by smooth algorithms to achieve an accuracy of 1% on nonsmooth problems . . . 165

22 Number of function evaluations required by nonsmooth algorithms to achieve an accuracy of 1% on nonsmooth problems . . . 166

23 Number of function evaluations required by nonsmooth algorithms to achieve an accuracy of 1% on smooth problems 168

24 Performance comparison of four methods for two-variable airfoil optimization 185

25 Performance comparison of four methods for four-variable airfoil optimization 187

26 Aircraft design variables and ranges 196

27 Aircraft design objective and constraints 197

28 Baseline versus optimized configuration found by `fBcEGO`, tabulated results 204

29 Comparison of optimized configurations generated by competing methods, tabulated results 207

30 Comparison of failed iterations and infeasible designs for each method 208

31 Data for Modified Langerman Problem 221

32 Some global minimizers of the Modified Langerman Problem 221

33 Global optimizers for Paviani problem 222

34 Data for the Hartman function 223

35 Data for Shekel's Foxholes 224

36 Data for the Shekel-m function 225

37 Some global minimizers for the Shekel-m function 225

38 Data for test problem N-16 232

39 Data for test problem N-19 235

LIST OF FIGURES

1 The Boeing 757 series and the Boeing Sonic Cruiser 2

2 Classical approach contrasted against the notional benefits of the "virtual product" approach 3

3 CPU hours required for a single run for some aerospace design tools . 4

4 Contour profiles of the design space as a function of wing area and thrust-to-weight ratio 7

5 Design space of an airfoil design problem 7

6 General framework for surrogate model-based global optimization and initial questions 14

7 Modeling accuracy of various surrogate models for the Branin function 18

8 Illustration of a stochastic process model 19

9 Illustration of the GP conditioning process 24

10 Illustration of the GP conditioning process with β marginalized . . . 25

11 Illustration of anisotropic and isotropic covariance functions 27

12 The effect of the number of observations on the likelihood 30

13 Log-likelihood as a function of hyperparameters 31

14 MLE fit compared with fully Bayesian fit 35

15 Bayesian penalization of unnecessarily complex and overly simple models 37

16 Dense posterior distribution 37

17 Test function used to demonstrate the proposed method along with dense hyperparameter posterior 40

18 Fully Bayesian model of the test function with initial prior 41

19 Fully Bayesian model of the test function with revised prior 42

20 Fully Bayesian model of the test function with final prior 43

21 Some of the favored modes generated by the proposed method 44

22 Comparison of proposed method with MLE 45

23 Illustration of surrogate model-based approach 51

24 Selected iterations of a GP-based P-algorithm on test function (B.11) 56

25 Graphical interpretation of the expected improvement criterion 59

26 Selected iterations of the EGO algorithm on test function (B.11) . . . 61

27 Selected iterations of the fully Bayesian EGO algorithm on test function (B.11) 64

28 Illustration of one-stage methods 65

29 Selected iterations of the osEGO algorithm on test function (B.11) . . 67

30 Selected iterations of the RBF-G algorithm on test function (B.11) . 72

31 Selected iterations of the CORS-RBF algorithm on test function (B.11) 75

32 First three iterations of DIRECT on a hypothetical problem 76

33 DIRECT's Pareto-based selection scheme 77

34 Performance of DIRECT on Branin function after $N = 20$ simplex gradients 77

35 Example of ℓ_1 nonsmooth penalty function 79

36 Performance of the constrained EGO algorithm 82

37 Graphical interpretation of the expected violation criterion 84

39 Regression and interpolation of drag polar data from XFOIL 6.94 for NACA 2432 airfoil 93

40 Fully Bayesian GP regression with and without reinterpolation 95

41 Regression and interpolation of drag polar data from XFOIL 6.94 for NACA 2432 airfoil using a fully Bayesian approach 96

42 Posterior distribution $p(\theta, \theta_n | y^{(k)})$ for airfoil problem 96

43 Methodology flowchart 107

44 Number of bound constrained test problems of dimension n 119

45 Number of nonlinearly constrained test problems of dimension n . . . 119

46 Size of nonlinearly constrained test problems in terms of equivalent number of inequality constraints 120

47 Test problem NS-3 121

48 Error between MATLAB's sqrt(x_0) and the Newton approximation to $\sqrt{x_0}$ 121

49 Noise corrupted quadratic function 122

50 Notional data profile 127

51 Test matrices for Experiment 1, Experiment 2.2, and Experiment 5 . 132

52 Test matrices for Experiment 2.1 and Experiment 3 133

53 Test matrices for Experiment 4 133

54 Effect of initial design on performance of algorithms 140

55 Results of bound constrained tests 143

56 Box plots of bound constrained tests with accuracy metric 144

57 Performance of competing algorithms on ten bound constrained problems with $n \leq 3$ over nine initial designs 148

58 Box plots of test results of ten bound constrained problems with $n \leq 3$ over nine initial designs 148

59 Results of nonlinearly constrained tests with accuracy metric 152

60 Box plots of nonlinearly constrained tests with accuracy metric 153

61 Results of nonlinearly constrained tests: equality constraint violation 154

62 Box plots of nonlinearly constrained tests with equality constraint violation 155

63 Results of nonlinearly constrained tests for all N 156

64 Results of noise corrupted tests 158

65 Box plots of noise corrupted tests with accuracy metric 159

66 Results of noise corrupted tests by budget and noise level 160

67 Performance of competing algorithms on nine nonsmooth problems with $n \leq 3$ over nine initial designs 163

68 Box plots of test results of nine nonsmooth problems with $n \leq 3$ over nine initial designs 164

69 Performance of nonsmooth algorithms on ten smooth problems with $n \leq 3$ over nine initial designs 167

70 Performance of `fBcEGO` with varying I on eleven bound constrained problems 170

71 Box plots of the performance of `fBcEGO` with varying I on eleven bound constrained problems 171

72 Orthogonal basis function airfoil parameterization 174

73 Surface pressure distributions of original and baseline airfoils 175

74 Performance of genetic algorithm modified to handle hard constraints 176

75 Some possible airfoils for the two-dimensional problem 178

76 Airfoil optimization design space in two dimensions 181

77 Performance of fBcEGO on the two-variable optimization problem . . 184

78 Contour plot of $10^4 \times c_d(w_2, w_3)$ with performance of fBcEGO versus DIRECT 185

79 Baseline versus optimized geometry for two-variable optimization . . 186

80 Performance of fBcEGO on the four-variable optimization problem . . 188

81 Performance of DIRECT on the four-variable optimization problem . . 189

82 Baseline versus optimized geometry for four-variable optimization . . 190

83 Surface pressure distributions of optimized airfoils 190

84 Bombardier CRJ-700 191

85 Mission profile for notional 70-passenger jet 192

86 Gross weight, landing field length, and block fuel as functions of wing area and thrust-to-weight ratio 193

87 Visualization of design space bounds by aircraft configuration 198

88 Baseline versus optimized configuration, overlay 203

89 Sensitivity of responses to variables at the optimum 205

90 Contour profiles of the design space as a function of wing area and thrust-to-weight ratio 206

91 Comparison of optimized configurations generated by competing methods207

92 Sampling of fBcEGO with N_1 DGS+FCP initial design 209

93 Sampling of fBcEGO with N_2 LHD+FCP initial design 209

94 Sampling of GA 210

95 Sampling of DIRECT 210

96 Number of samples versus total analytical time of fBcEGO on FLOPS problem 217

97 Application BMC to the integral $\overline{y}_p = \int y(x)p(x)dx$ with $y(x) = \text{sinc}x$ and $p(x) = \mathcal{N}(0, 1)$ 245

98 Convergence of BMC and SMC versus sample size for two functions . 246

LIST OF SYMBOLS AND ABBREVIATIONS

$()^\ell$	Lower bound
$()^u$	Upper bound
$(\cdot)_+$	Maximum of the arguments and 0
$\lvert\cdot\rvert$	Determinant (matrices), absolute value (scalars), cardinality (set)
β	$p \times 1$ vector of regression coefficient
$\delta(\cdot)$	Delta distribution
δ_{ii}	Kronecker delta
ϵ	Noise (§2.2), prescribed accuracy
ϵ_n	Relative noise level in noise corrupted functions
ϵ_n	Noise term
$\Gamma(\cdot)$	Gamma function
$\hat{\theta}$	Estimator of θ
$\hat{c}_i(\cdot)$	Mean function of $C_i(\cdot)$
$\hat{y}(\cdot)$	Model of $y(\cdot)$; mean function of $Y(\cdot)$ in the case of GP models
λ	$k \times 1$ vector of RBF coefficients (§2.3)
λ_i	RBF coefficients (§2.3)
$\lceil\cdot\rceil$	Ceiling operator
$\lfloor\cdot\rfloor$	Floor operator

log	Natural logarithm
$\Bbbk(\cdot)$	Covariance function; the form $\Bbbk(\cdot, \cdot)$ may also be used to explicitly define the two points of interest
$\mathbb{E}[\cdot]$	Expectation
$\mathbb{R}_+$	Set of positive real numbers
$\mathbf{\Psi}$	Covariance matrix of a GP corrupted by noise
$\mathbf{h}$	The vector $\lvert x - x^\star \rvert$ between two points x and $x^\star$
$\mathbf{I}$	Identity matrix
$\mathbf{K}$	$k \times k$ covariance matrix with elements $\Bbbk(x^{(i)}, x^{(j)})$, $i, j = 1, \ldots, k$
$\mathbf{k}_0$	$k \times 1$ covariance vector between x_0 and $x^{(i)}$, $i = 1, \ldots, k$
$\mathcal{B}$	Set of events in Ω
$\mathcal{D}$	Set of design points $\{x^{(i)}\}, i = 1, \ldots, k$
$\mathcal{K}_\nu(\cdot)$	Modified Bessel function of order ν
$\mathcal{N}(\mu, \Sigma)$	Normal distribution with mean μ and covariance matrix Σ
$\mathcal{S}$	Set of observations $\{y(x^{(i)})\}$, $i = 1, \ldots, k$
$\mathfrak{p}$	Number of polynomial regression terms in a RBF interpolation
$\lVert\cdot\rVert_p$	Vector p-norm
ν	Smoothness hyperparameter of the Matérn covariance function
Ω	Sample space
Φ	$k \times k$ matrix of RBFs $\phi(\cdot)$ evaluated at $\lVert x^{(i)} - x^{(j)} \rVert_2$, $i, j = 1, \ldots, k$ (§2.3)

$\Phi(\cdot)$	Standard normal distribution function (§3.2.2)
$\phi(\cdot)$	Radial basis function (§2.3), standard normal density function (§3.2.2)
σ^2	Variance
σ_n^2	Noise variance
P_0	Probability measure, i.e., $\mathrm{P}_0 : \mathcal{B} \rightarrow [0,1]$
θ_h	Lengthscale hyperparameter of a covariance function, $h = 1, \ldots, n$
b	$d \times 1$ vector of polynomial regressor coefficients (§2.3)
b_i	Polynomial regressor coefficients (§2.3)
$C_i(\cdot)$	GP model for $c_i(\cdot)$
$c_i(\cdot)$	Scalar constraint function, i.e., $c_i : \mathbb{R}^n \rightarrow \mathbb{R}$
F	$k \times p$ matrix of regressors having (i,j)th element $f_j(x^{(i)})$ for $i = 1, \ldots, k$, $j = 1, \ldots, p$
$f(\cdot)$	Nonlinear regression function
g	Generalized EI parameter
$I(\cdot)$	Improvement function
k	Number of observations
m	Number of inequality constraints
n	Leading dimension of x
P	$k \times k$ matrix of polynomial regressors $p_j(x^{(i)})$, $i = 1, \ldots, k$ (§2.3)
p	Number of regression coefficients (§2.2)

$p(\cdot)$	Polynomial regressor (§2.3), probability function
p_h	Power hyperparameter of an exponential covariance function, $h = 1, \ldots, n$
$s^2(\cdot)$	Posterior variance of $Y(\cdot)$ conditioned on $y^{(k)}$
$T(\cdot)$	Student t CDF
$t(\cdot)$	Student t p.d.f
$v(h;\theta)$	Univariate Matérn covariance function
x	$n \times 1$ vector of design variables
$x^{(i)}$	Sample point i, i.e., the point x where observation i is to be made
$Y(\cdot)$	Random process; the form $Y(\cdot, \Omega)$ may also be used to explicitly indicate the sample space Ω
$y(\cdot)$	Scalar objective function, i.e., $y : \mathbb{R}^n \to \mathbb{R}$
y^*	Global minimum of $y(\cdot)$
y^T	Target value
$y^{(k)}$	$k \times 1$ vector of observations
y_0	Unknown output value of $y(\cdot)$ at x_0, i.e., $y_0 \triangleq y(x_0)$
$y_{\min}$	Minimum feasible sample
$z(\cdot)$	Noise corrupted observation of $y(\cdot)$
`DIRECT`	*Di*viding *Rect*angles
`fBcEGO`	Fully Bayesian Constrained Efficient Global Optimization
a.s.	Almost surely

CFD	Computational fluid dynamics
COBYLA	Constrained Optimization BY Linear Approximation
CORS	Constrained Optimization using Response Surfaces
CPS	Corner point strategy
DGS	Deterministic global solver
EGO	Efficient Global Optimization
EI	Expected improvement
FCP	Fractional corner point design
FEA	Finite element analysis
GP	Gaussian process
MAP	Maximum *a posteriori*
MLE	Maximum likelihood estimation
p.d.f.	Probability density function
p.m.f.	Probability mass function
RBF	Radial basis function
$\mathrm{Tr}(\cdot)$	Trace of a matrix

SUMMARY

The design of unconventional aircraft requires the early use of advanced computational tools such as computational fluid dynamics and finite element methods in order to predict the performance of the designs outside the range of contemporary experience. The primary technical challenge that precludes the use of advanced computational tools in early design is the cost in CPU hours required for one design evaluation. The computational expense, the approximation error in the solutions due to the convergence of numerical simulations, the unknown convexity and continuity properties and bounds of the design space, and the possible existence of hard constraints which can result in failed evaluations make traditional optimization techniques ill-suited for design optimization of unconventional aircraft. Current methods for incorporating advanced computational tools into early design phases are inadequate due to the restricted budgets that are common in early design. This motivates the need for a robust and efficient global optimization algorithm.

This research presents a novel surrogate model-based global optimization algorithm to efficiently search challenging design spaces for optimum designs. The algorithm, named `fBcEGO` for fully Bayesian constrained efficient global optimization, constructs a fully Bayesian Gaussian process model through a set of evaluations of an unknown function and then uses the model to make new observations in areas where the expected improvement over the current best value is relatively large. A challenge in the construction of the fully Bayesian Gaussian process model is the selection of the prior distribution placed on the model hyperparameters. Previous work has employed static priors, which may not capture a sufficient number of interpretations of the data to make any useful inferences about the underlying function. Thus, the first contribution of this research is an iterative method that dynamically assigns hyperparameter priors by exploiting the mechanics of Bayesian penalization in order to marginalize a sufficient number of interpretations. The method corrects the inadequacies of likelihood-based approaches, which can result in poor inferences of the unknown function when function evaluations are expensive and therefore scarce. By fitting many models through the observations and viewing them under the Bayesian methodology, much more can be learned about the hyperparameters than from likelihood-based approaches, which compute point values for the hyperparameters. Marginalizing the uncertainty of the hyparameters into the fully Bayesian model causes the search phase to be more global initially when compared with likelihood-based approaches, which is beneficial because a region containing the global minimum is less likely to be overlooked. The algorithm is extended to general nonlinear programs through the use of a fully Bayesian constrained expected improvement criterion.

`fBcEGO` is incorporated into a methodology that reduces failed cases, infeasible designs, and provides large reductions in the objective function values of design problems. Four sets of algebraic test problems, including bound constrained, noise corrupted, nonsmooth, and nonlinearly constrained problems, were compiled in order to test `fBcEGO`'s abilities in comparison with state-of-the-art methods from the literature. `fBcEGO` is shown to solve up to three times as many nonlinearly constrained problems than the competing methods for a given function evaluation budget. The methodology is applied to an airfoil section design problem and a conceptual aircraft design problem. The methododology obtains the largest reduction in the takeoff gross weight of a notional 70-passenger regional jet versus competing design methods. Additional contributions include research on adaptive initial designs and methods for handling hard constraints.

CHAPTER I

MOTIVATION

1.1 Design Of Advanced Aircraft Concepts

The development of a new aircraft can be categorized into three major phases: conceptual, preliminary, and detailed design. Design is iterative in nature, with each iteration or spiral refining the previous solution. Conceptual design of derivative aircraft is typically performed using simplified physics-based analyses and empirical design tools based on historical data for each of the design disciplines [96, 95]. These tools may be unable to predict the behavior of unconventional designs outside the range of contemporary experience. Figure 1 shows examples of two derivative aircraft, the Boeing 757-200 and 757-300, and an advanced concept, the Boeing Sonic Cruiser. The 757 family is similar to previous commercial aircraft, thus previous knowledge can be used to predict the performance of derivative designs in early design stages. However, the performance of the Sonic Cruiser, which is meant to travel just below Mach 1.0 [2], cannot be adequately predicted using previous knowledge. Thus, the design of advanced or unconventional aircraft requires early use of high-fidelity physics-based tools such as computational fluid dynamics (CFD) and finite element analysis (FEA).

Computational methods in the aerospace industry have revolutionized design [52, 63, 68, 57, 30]. CFD and FEA have joined the wind tunnel, structural limits test, and flight test as primary tools of the trade. In contrast with full-scale prototype testing, computational methods can inexpensively produce simulations leading to an improved understanding necessary for design. Effective use of computational methods at the appropriate design stage is a key ingredient in the successful design of modern

Figure 1: The Boeing 757 series, a family of derivative aircraft (left) and the Boeing Sonic Cruiser (right) [www.boeing.com].

aircraft.

Perhaps the greatest utility of advanced computational tools is through their use in "inverse design", or optimization, predicting the necessary geometric or system changes to optimize a certain cost function. This feature is of great value in aerospace design, since small changes in shape or size may lead to significant performance gains due to the highly coupled and nonlinear nature of the underlying physics. The computational expense of high-fidelity tools often precludes their use in early design, where decisions are made that determine up to 80% of the life-cycle cost of a product [100]. However, it is in these early stages where a wide range of unconventional designs are considered and global rather than local optimization can play a critical role. Incorporating these tools in early design can lead to better decision making due to increased fidelity and reduced uncertainty of the design space.

Figure 2 shows the notional benefits of this idea. By incorporating high-fidelity design tools into early design, a "virtual product" is created. A virtual product is a "high-fidelity mathematical/numerical representation of the physical properties and the functions of a product" [120]. Some design milestones, e.g., the first wind tunnel test, are shown in Figure 2; these may be achieved earlier through the use of high-fidelity tools in early design. However, the cost of even a few hundred runs of these tools (a modest number in optimization) can be prohibitive for early design, where resources are limited and a quick turn-around time is imperative. Figure 3 shows some tools that are utilized in advanced design and the typical CPU time required per run. Computational tools currently used in conceptual design, e.g., the panel code XFOIL, are characterized by short CPU run times on the order of seconds or minutes and typically employ linearized physics. Tools used in detailed design, e.g., the three-dimensional RANS solver `OVERFLOW`, may require hundreds or thousands of CPU hours to generate a single solution due to the increased fidelity that is required. Ideally, there is some method that will enable designers to exploit the benefits of advanced computational tools under the budget constraints imposed in early design. Thus, efficient global optimization algorithms have been identified as key enablers in the early design of advanced or unconventional aircraft.

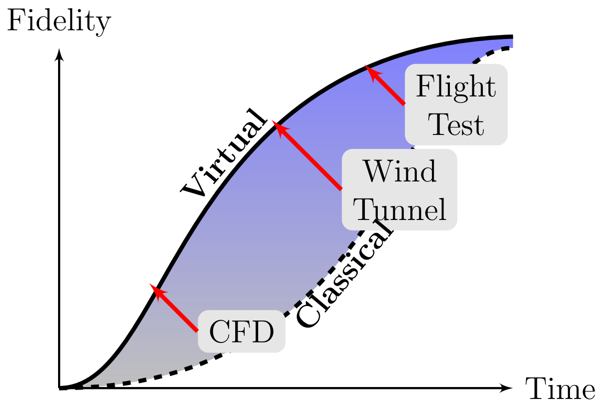


Figure 2: Classical approach contrasted against the notional benefits of the "virtual product" approach [100].

Efforts to employ high-fidelity analyses in aerospace conceptual design date back to the 1960s and 1970s [63] where designers developed polynomial response surfaces to datasets generated by designs of experiments and searched the design space by minimizing the response surface; this method continues to be used today [73]. However, the need to explore globally, the increasing complexity of design, and the often unknown design variable bounds render the classic response surface methodology inefficient for design optimization of advanced or unconventional aircraft. This has led to

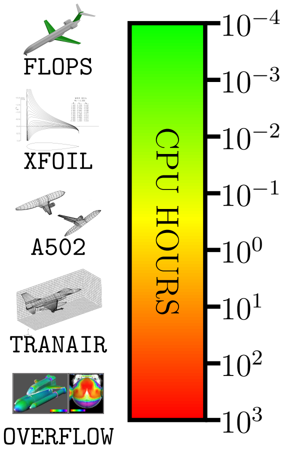


Figure 3: CPU hours required for a single run for some aerospace design tools [52, 68, 32, 102].

the development of advanced surrogate model-based global optimization algorithms that search for global solutions to design problems stated as nonlinear programs [54, 57, 39].

With the development of nonlinear programming [75, 117], the engineer gained the necessary mathematical tools to automatically reduce the design space or completely specify an optimal design subject to a set of requirements. A nonlinear program is a problem in the form:

$$\begin{aligned} \underset{x\in\mathbb{R}^n}{\text{minimize}} \quad & y(x) \\ \text{subject to} \quad & c_i(x) = 0, \qquad i \in \mathcal{E} \\ & c_i(x) \geq 0, \qquad i \in \mathcal{I} \\ & x^\ell \leq x \leq x^u \end{aligned} \tag{1.1}$$

where $x \in \mathbb{R}^n$ is the vector of design variables, $y : \mathbb{R}^n \to \mathbb{R}$ is the objective function, $c_i : \mathbb{R}^n \to \mathbb{R}$ are constraint functions, $\mathcal{E}$ and $\mathcal{I}$ are disjoint index sets of equality and inequality constraints, respectively, and x^u and x^ℓ are vectors of upper and lower bounds, respectively, on x. This problem can be recast, without loss of generality, in the following form, which is used in the remainder of this document:

$$\begin{aligned} \underset{x \in A \subset \mathbb{R}^n}{\text{minimize}} \quad & y(x) \\ \text{subject to} \quad & c_i(x) \geq 0, \quad i = 1, \ldots, m \end{aligned} \tag{NLP}$$

where $c_i : \mathbb{R}^n \to \mathbb{R}$ are m inequality constraint functions, and $A = \{x | x^\ell \leq x \leq x^u\}$. Problem (NLP) is completely general because an equality constraint $c_i(x) = 0$, can be written as an equivalent two-sided inequality constraint as follows:

$$c_i(x) = 0 \Longleftrightarrow \begin{cases} c_{i+m}(x) = c_i(x) + \delta \geq 0 \\ c_{i+m}(x) = \delta - c_i(x) \geq 0 \end{cases} \tag{1.2}$$

where δ is a relaxation parameter that controls the degree of satisfaction of $c_i(x) = 0$, i.e., for $\delta = 0$, $c_i(x) = 0$ is satisfied exactly.

1.2 Technical Challenges

The computational tools depicted in Figure 3 are subject to a set of technical challenges that is common to many design problems in the literature (see Table 1). The technical challenges arise from both the design code and the unfamiliarity of the design space of unconventional designs. The technical challenges are listed below.

Technical Challenges

1. Utilization of high-fidelity models makes the design codes computationally expensive, i.e., $y(\cdot)$ is a result of a computationally expensive simulation.
2. Only the objective and constraint function values are reported at a point x, i.e., the code is a “black-box” executable.
3. Mild deterministic noise is present in the problem due to the convergence of numerical simulations.

4. The problem is nonconvex or has unknown convexity properties.

5. The problem has unknown variable bounds.

6. The size of the feasible region is unknown, i.e., it is not known if any feasible designs exist given the constraints and variable bounds.

7. The problem may be nonsmooth, i.e., C^0, due to the utilization of different models in different regions of the design space.

8. Existence of hard or hidden constraints may cause the code to crash if x violates some constraint, e.g., geometric. The computational expense of a failed evaluation is of the same order of magnitude as a successful evaluation.

9. Disconnected feasible regions may be present.

10. Due to 1, 2, 3, 7, 8, and 9, neither analytical nor numerical derivatives are available.

11. Low-fidelity models to guide the high-fidelity optimization are either inaccurate or unavailable.

Figures 4 and 5 illustrate some of the technical challenges. Figure 4 shows the two-dimensional design space of notional aircraft from Chapter 9. This problem exhibits hard constraints, indicated by the hatched areas, and only 3.4% of the full-dimensional design space is feasible with respect to the inequality constraints and hard constraints. Figure 5 shows the design space of an airfoil optimization problem from Chapter 9 as a function of two geometric parameters. This problem exhibits hard constraints and code failures interior to the feasible region, as well as disconnected feasible regions. For both problems, the variable bounds were unknown and had to be assumed or estimated from a suitable method. It is clear that traditional optimization tools are ill-suited to solve such problems. The development of a novel global optimization algorithm is motivated by these technical challenges.

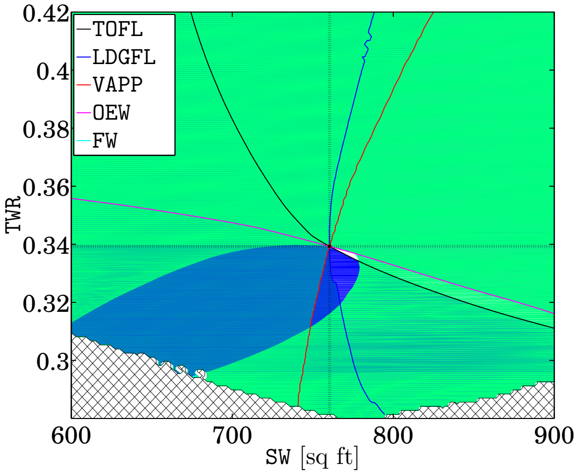


Figure 4: Design space of notional 70-passenger regional jet; takeoff gross weight as a function of wing area `SW` and thrust-to-weight ratio `TWR`.

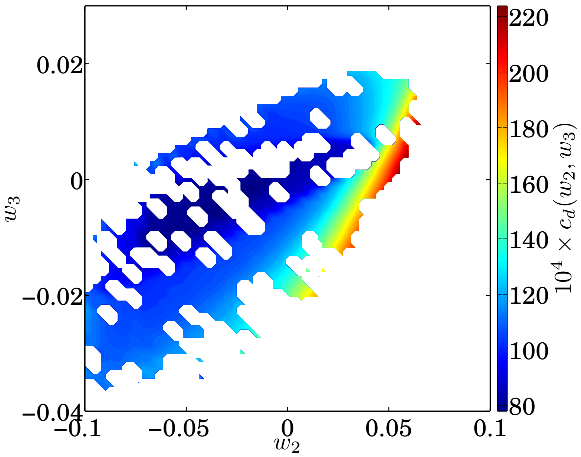


Figure 5: Design space of an airfoil design problem; sectional drag as a function of geometrical parameters.

Table 1: Computationally expensive black-box optimization problems from the literature.

Problem	Reference
Integrated circuit, automotive design problem	Jones et al. [56]
2D airfoil design, 3D wing design, wing and flap track fairing	Forrester [40]
Electro-mechanical actuator	Messine [66]
2D airfoil design, 3D wing design	Alexandrov et al. [4]
Boeing wing planform design	Audet and Jr. [7], Audet et al. [8]
Helicopter rotor blade design	Booker et al. [18], Conn et al. [25]
Heave motion of a ship in head seas	Campana et al. [23]
Containership design	Sherali and Ganesan [109]
Metal spinning	Henkenjohann and Kunert [48]
Oilshale pyrolysis, nonlinear continuous stirred reactor	Meyer et al. [67]
Circuit simulator	Sacks et al. [103]
Multiobjective vehicle system analysis	Sasena et al. [107]
Automotive piston design	Schonlau et al. [108]
Parametric vehicle controller	Villemonteix et al. [119]
Automotive intake port	Villemonteix et al. [119]
Centrifugal compressor blade design	Berghen [11]
Electricity meter, vibromotor, shock absorber, magnetic beam deflection system, small aperture coupling between a rectangular waveguide and a microstrip line, large scale integration, circuit board etching, pigment compounds, electromechanical adsorption, immunological model, nonstationary queuing system, Steiner problem	Mockus [69]

1.3 Research Focus & Approach

The investigation of the technical challenges in §1.2 has lead to the following primary research objective:

Primary Research Objective

Develop a surrogate model-based global optimization algorithm for the general nonlinear programming problem

$$\begin{aligned} \underset{x\in A\subset\mathbb{R}^n}{\text{minimize}} \quad & y(x) \\ \text{subject to} \quad & c_i(x) \geq 0, \quad i = 1\ldots, m \end{aligned} \tag{NLP}$$

where $x \in \mathbb{R}^n$ is the vector of design variables, $y : \mathbb{R}^n \to \mathbb{R}$ is the deterministic objective function, $c_i : \mathbb{R}^n \to \mathbb{R}$ are m deterministic inequality constraint functions, and and $A = \{x|x^\ell \leq x \leq x^u\}$.

The following primary research question will guide the literature review in the sequel:

Primary Research Question

Within the context of surrogate model-based global optimization, what type of algorithm can solve (NLP) more efficiently than the state-of-the-art methods found in the literature?

The problem discussed thus far is difficult from two perspectives: a theoretical perspective and a practical algorithm design perspective. The theoretical difficulties of global optimization are well known. While a local minimum can be mathematically characterized using the Karush-Kuhn-Tucker conditions [117], there are no conditions to characterize a global minimum except when the problem can be shown to be convex. The only other way to prove that a local minimum is a global minimum is to identify all local minima. Thus, when global optimization is possible, it is generally exhaustive, leading to exponential worst-case time complexity for continuous problems and factorial complexity for discrete problems. Few classes of problems exist where exhaustive search is practical, e.g., polynomial programming problems [74] and small-scale combinatorial problems; in general, heuristic approaches must be used to find solutions that are "good enough." Nevertheless, the following quote, which relates directly to design problems, describes why global optimization is still important:

> "Often, it may be of considerable interest to know at least whether there exists any better solution to a practical problem than a given solution that has been obtained for example by some local method." [115]

The research problem is also difficult from an algorithm design perspective. There is little practical guidance available for the design of heuristic algorithms, but these can be designed through a combination of literature review and numerical experience. This is in contrast to rigorous algorithms that are based on mathematical conditions. Table 2 demonstrates the combinatorial aspect of algorithm design. Some alternatives are listed that may be used to address the technical challenges in §1.1. The table is not exhaustive. A literature review [75, 117, 39] of available methods leads to the elimination of a large number of alternatives, which have been grayed out in Table 2. This process is described next.

The alternatives that can be used to address the black-box technical challenge are reduced by noise and expense considerations. For example, the approximation error present in design codes makes finite difference estimates of the gradients impossible, while direct methods and metaheuristics can be eliminated due to expense considerations. For global search, expense considerations can be used to eliminate

Table 2: Matrix of alternatives for the technical challenges.

Challenge	Alt. 1	Alt. 2	Alt. 3	Alt. 4	Alt. 5
Black-box	Surrogate model	FD derivatives	Direct method	Metaheuristics	
Global search	Metaheuristics	Clustering	Branch & bound	Probabilistic criterion	
Local search	Trust-region	Line search	Nelder-Mead	Local model	Probabilistic criterion
Constraint handling	SQP	Interior-point	Filter	Penalty	Probabilistic criterion
Computational expense	Budget	Loose termination criteria	Reduced-order models	Stochastic process	Use all previous data
Nonconvexity	Trust-region	Metaheuristics	Multi-start	Adaptive surrogate model	
Noise & smoothness	Surrogate model	Direct method	Trust-region	Metaheuristics	
Hard constraints	Assign value	Discard			

metaheuristics and clustering, while branch & bound methods are eliminated because rigorous lower bounds cannot be obtained from black-box design codes that only output function evaluations. For local search, the Nelder-Mead method is eliminated due to expense considerations, while trust-region methods, line search methods, and local models are eliminated because these methods are local and become expensive under any globalization strategy, e.g., multistart. The classical trust region methods also require gradient information, which cannot be obtained here. Expense considerations can be used to eliminate alternatives for constraint handling, nonconvexity, and noise & smoothness. Finally, for failed iterations resulting from hard constraints, discarding the failed iterate is not an economical approach. The remaining alternatives indicate that a probabilistic surrogate model-based method combined with some additional techniques to handle the remaining technical challenges can be used to achieve the primary research objective.

The general framework for such an algorithm [54] is outlined in Algorithm 1. The general idea is to evaluate the deterministic objective function $y(\cdot)$ on some initial set of points $\mathcal{D}^{(1)} \subset A$ which satisfies the bound constraints of problem (NLP). A surrogate model is built from the sampled points, which is used to derive an infill sampling criterion (ISC). The ISC attempts to balance local and global search in some manner in order to identify promising areas of the design space where the global minimum is likely to occur. The maximizer of the ISC then serves as the observation site for the next function evaluation. The sample sets are then updated and the process iterates until some convergence criterion is met. A key difference between this type of approach and the classical response surface methodology in design [73, 60] is that there are no goodness-of-fit tests. This is because the goal of surrogate model-based global optimization is to focus resources in areas of the design space where the global minimum is likely to occur, and not to develop a globally-accurate surrogate model, although there is no limitation that prevents this alternative goal from being achieved through an alternative ISC.

A flowchart version of Algorithm 1 is presented in Figure 6. The major details have been left unspecified, as this will form the core of the research. A set of initial questions corresponding to each box in the flowchart is presented in Figure 6. These will be refined into research questions in Chapter 7, which will be further developed into hypotheses and experiments. Chapter 2 reviews some function approximation techniques that have been used in surrogate model-based methods. The first contribution, a method for assigning hyperparameter priors for fully Bayesian Gaussian process models, is described here. Chapter 3 contains an in-depth analysis of existing ISCs for computationally expensive black-box optimization of bound constrained problems and includes a derivation of a fully Bayesian ISC that is used in this research. Chapter 4 reviews methods for handling nonlinear constraints within the context of surrogate model-based global optimization. Chapter 5 discusses additional considerations that are necessary for addressing the remaining technical challenges. Chapter

6 combines the method for assigning hyperparameter priors, the fully Bayesian ISC, and the additional techniques for addressing the technical challenges into a methodology for global optimization of computationally expensive design problems. Chapter 7 outlines a testing plan and develops formal research questions and hypotheses that will be used to test the proposed algorithm against existing methods. Test problems from Chapter 7 are documented in Appendix A. Chapter 8 presents the results of the tests from Chapter 7. Finally, Chapter 9 presents the implementation results of the proposed methodology on two aircraft-related design problems.

Algorithm 1 General framework for bound constrained surrogate model-based global optimization.

```
Select initial design $\mathcal{D}^{(1)} = \{x^{(1)}, \dots, x^{(k)}\} \subset A$ and compute the initial sample set $\mathcal{S}^{(1)} = \{y(x^{(1)}), \dots, y(x^{(k)})\}$
Set $y_{\min} \leftarrow \infty$, $i \leftarrow 1$
while not converged do
    Fit model to $\mathcal{D}^{(i)}$ and $\mathcal{S}^{(i)}$
    Compute the next sample point by solving
        $$x^{(k+1)} = \arg \max_{x \in A} ISC(x)$$
    Update: $\mathcal{D}^{(i+1)} \leftarrow \mathcal{D}^{(i)} \cup x^{(k+1)}$, $\mathcal{S}^{(i+1)} \leftarrow \mathcal{S}^{(i)} \cup y(x^{(k+1)})$, $i \leftarrow i+1$, $k \leftarrow k+1$
    $y_{\min} \leftarrow \min_{1 \leq i \leq k} y(x^{(i)})$
end while
```

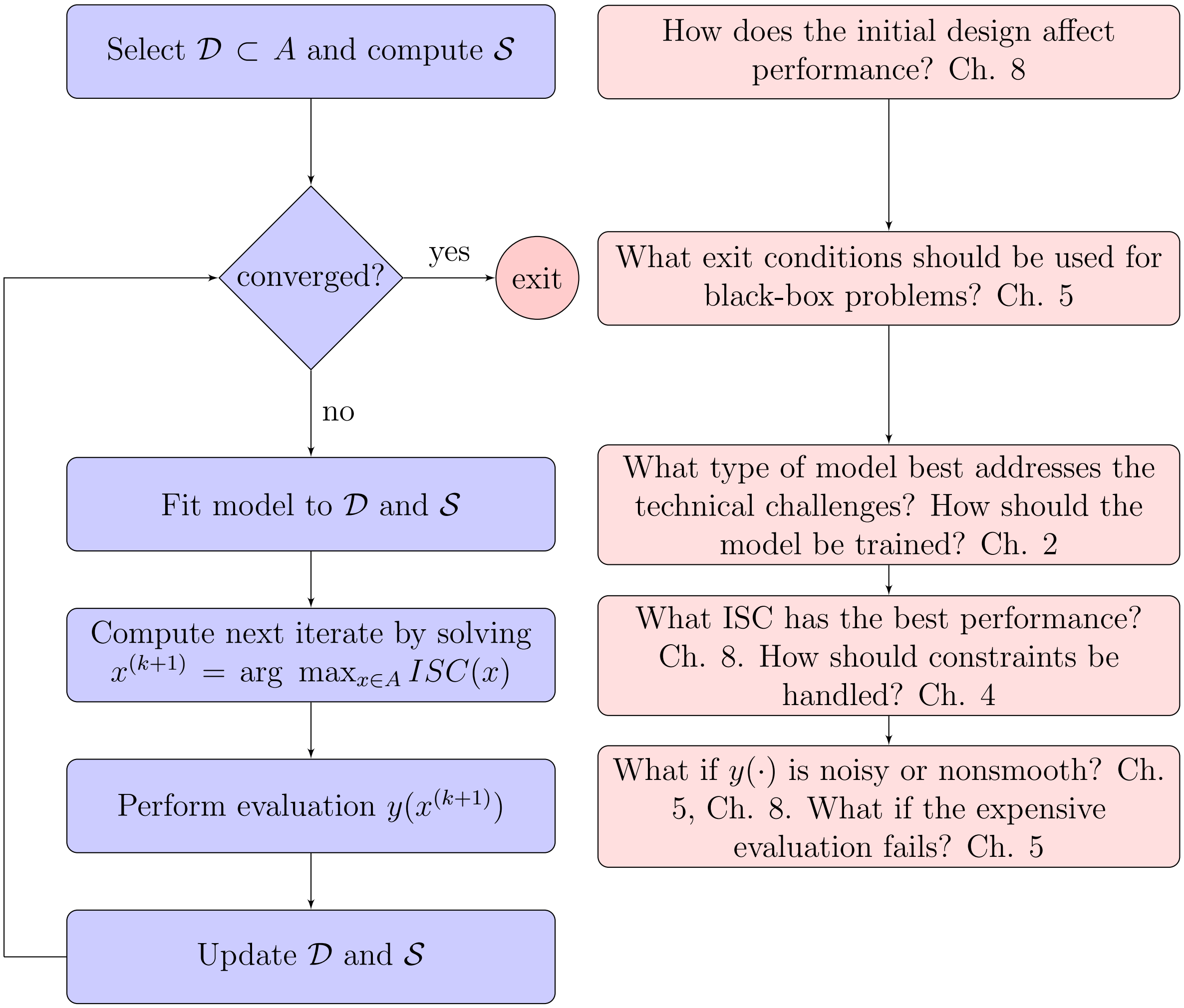

Select $\mathcal{D} \subset A$ and compute $\mathcal{S}$
converged?
yes
exit
no
Fit model to $\mathcal{D}$ and $\mathcal{S}$
Compute next iterate by solving $x^{(k+1)} = \arg\ \max_{x \in A} ISC(x)$
Perform evaluation $y(x^{(k+1)})$
Update $\mathcal{D}$ and $\mathcal{S}$
How does the initial design affect performance? Ch. 8
What exit conditions should be used for black-box problems? Ch. 5
What type of model best addresses the technical challenges? How should the model be trained? Ch. 2
What ISC has the best performance? Ch. 8. How should constraints be handled? Ch. 4
What if $y(\cdot)$ is noisy or nonsmooth? Ch. 5, Ch. 8. What if the expensive evaluation fails? Ch. 5

Figure 6: General framework for surrogate model-based global optimization and initial questions.

CHAPTER II

FUNCTION APPROXIMATION TECHNIQUES

2.1 Selection Of Surrogate Model

The selection of the surrogate model is presented first, followed by the mathematical details. The following qualitative criteria, derived from the technical challenges in §1.2, were used to determine which surrogate model would be used in the proposed algorithm in Chapter 6:

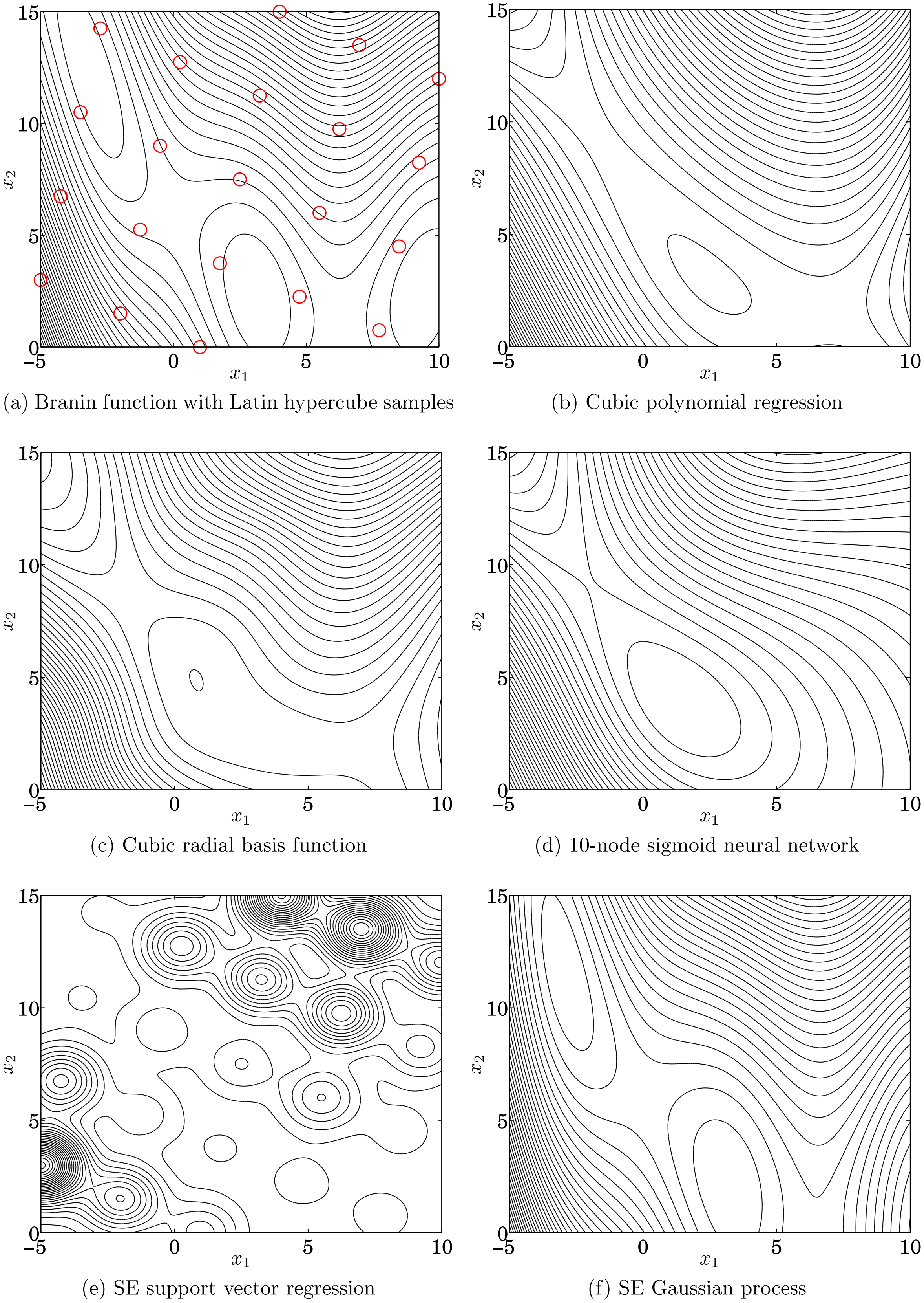
1. **Flexibility:** How well does the model capture nonlinearities?
2. **Effort:** How much computational effort is required to train the model?
3. **Sample size:** How many observations are required to construct an adequate model?
4. **Interpretation:** What interpretation, if any, does the model provide?
5. **Uncertainty estimate:** Does the model provide an uncertainty estimate?
6. **Noise:** Can the model handle noise corrupted observations?

The surrogate models are graded in Table 3. Polynomial models are not flexible enough and if their degree is increased too much the models will begin to interpolate noise. Polynomial regression requires a large number of observations and polynomial interpolation has other restrictions. For example, the sample sites must meet some geometric requirements to ensure that the interpolating model is well conditioned [26]. This is highly restrictive to the search algorithm. Artificial neural networks are more flexible than polynomial regression but still require a large number of observations.

Support vector regression (SVR) models do not use all observations, which is undesirable in the context of expensive problems [57]. Radial basis function (RBF) models are acceptable and are used in some of the algorithms reviewed. The RBF spline interpretation is useful in developing a search criterion within a one-stage framework, e.g., [44]. Stochastic process models (used in the context of surrogate modeling) are a type of RBF model but the basis functions are parameterized via the hyperparameters. The basis functions for RBFs are fixed in advance while the basis functions for stochastic process models are tuned to the observations. Thus, stochastic processes can be expected to exhibit superior accuracy over RBFs. In addition, the Bayesian interpretation enables derivation of a probabilistic search criterion and also provides an uncertainty estimate. Finally, stochastic process models can either interpolate or regress data through a minor modification. Thus, stochastic process models, and in particular Gaussian process (GP) models, are selected as the surrogate model. A demonstration of each surrogate model is illustrated in Figure 7.

Table 3: Relative capabilities of surrogate models.

Model type	Flexibility	Effort	Sample size	Interpretation	Uncertainty est.	Noise
Polynomial regression	Low	Low	High	–	No	Yes
Polynomial interpolation	Low	Low	High	–	No	No
Artificial neural network	Moderate	High	High	–	No	Yes
Support vector regression	High	High	Low	–	No	Yes
Radial basis function	High	Low	Low	Spline	No	No
Stochastic process	High	High	Low	Bayesian	Yes	Yes



(a) Branin function with Latin hypercube samples

(b) Cubic polynomial regression

(c) Cubic radial basis function

(d) 10-node sigmoid neural network

(e) SE support vector regression

(f) SE Gaussian process

Figure 7: Modeling accuracy of various surrogate models for the Branin function.

2.2 Gaussian Process Models

The material in this section is based on [104, 94, 57].

2.2.1 Mathematical Background

A Gaussian process (GP) is a collection of random variables, any finite number of which are jointly Gaussian [94]. A Gaussian process model is a model which views a deterministic response $y(\cdot)$ as a realization $y(\cdot,\omega)$, $\omega \in \Omega$ of a Gaussian random process $Y(\cdot,\Omega)$ defined on some probability space $(\Omega, \mathcal{B}, \mathrm{P}_0)$. The concept of a random process is analogous to a random variable. A random variable is a variable that assigns some nonnegative value called a probability to a certain outcome of an experiment, whereas a random process assigns a function drawn from some prior process to the outcome. Figure 8 illustrates the process.

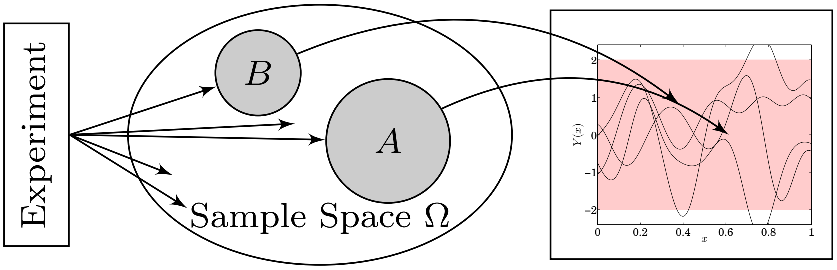


Figure 8: Illustration of a stochastic process model. An experiment generates some events A and B in the sample space Ω and each event is assigned some function generated from a prior process.

The outcome $y(\cdot,\omega)$ of the experiment may be approximated through a linear regression plus GP:

$$\begin{aligned} Y(x) &= \sum_{h=1}^{p} \beta_h f_h(x) + Z \\ &= \mathbf{f}^T \beta + Z \end{aligned} \tag{2.1}$$

where β_h are the unknown hyperparameters to be estimated from a sample set $\mathcal{S} = \{y(x^{(1)}), \ldots, y(x^{(k)})\}$, $f_h(\cdot)$, $h = 1, \ldots, p$, are known nonlinear functions, and $Z \sim \mathcal{N}(0, s^2)$ is a zero mean GP prior with variance $s^2(\cdot)$ of known form but unknown hyperparameters, also to be estimated from the sample set $\mathcal{S}$.

For a deterministic model, it is assumed that the lack of fit of the linear regression term in equation (2.1) to $\mathcal{S}$ is entirely due to an incomplete set of regression terms in x such that the modeling error between any two points x and $x^\star$ is dependent upon some metric $d(x, x^\star)$ that is zero for $x = x^\star$. A stationary covariance function is employed for this reason. Loosely speaking, a stationary covariance function is a function of $\mathbf{h} = |x - x^\star|$ and is thus invariant to translations in x. In general, a covariance function cannot be an arbitrary function of $\mathbf{h}$; see [104, §2.3.3], [94, §4.1] for the mathematical properties it must satisfy.

A popular covariance function is the anisotropic power exponential class

$$\Bbbk(\mathbf{h}) = \theta_0 \exp\left(-\frac{1}{2}\sum_{h=1}^{n}\left|\frac{h_h}{\theta_h}\right|^{p_h}\right) \tag{2.2}$$

with $\theta_0, \theta_h > 0$ and $0 < p_h \leq 2$. The hyperparameter θ_0 controls the magnitude (or variance) while θ_h and p_h control the nonlinearity and differentiability of the sample paths from $Y(\cdot)$, respectively. That is, a small value of θ_h indicates that the correlation for a pair of inputs decreases rapidly over a small distance, hence $Z(x)$ appears more like white noise. For powers $p_h < 2$, the sample paths are theoretically nondifferentiable. For $p_h = 2$, the power exponential correlation function becomes the infinitely differentiable Gaussian covariance function, also known as the squared exponential (SE) covariance function. The isotropic form of the power exponential covariance function is obtained by setting $\theta_1 = \theta_2 = \cdots = \theta_n$ and $p_1 = p_2 = \cdots = p_n$. For any GP with an isotropic covariance function, a point x has the same covariance with every point on a hypersphere about x. Other classes of covariance functions exist [104, 94] but the power exponential class is the most general and allows many shapes to be modeled when no continuity assumptions are made about $y(\cdot)$.

In particular, the Matérn class of covariance functions may be used to incorporate the continuity assumptions of $y(\cdot)$. Define for the univariate case:

$$v(h;\theta) = \frac{2^{1-\nu}}{\Gamma(\nu)}\left(\frac{2\sqrt{\nu}\,|h|}{\theta}\right)^{\nu}\mathcal{K}_\nu\left(\frac{2\sqrt{\nu}\,|h|}{\theta}\right) \tag{2.3}$$

where $\Gamma(\cdot)$ is the Gamma function and $\mathcal{K}_\nu(\cdot)$ is the modified Bessel function of the second kind of order ν and $\nu, \theta > 0$. Products of (2.3) can be used for modeling n-dimensional input responses. In this case, the family might include dimension-specific scale and smoothness hyperparameters, i.e.:

$$\mathbb{k}(\mathbf{h}) = \theta_0 \prod_{h=1}^{n} \upsilon(h_h; \theta_h) \tag{2.4}$$

$$= \theta_0 \prod_{h=1}^{n} \frac{2^{1-\nu_h}}{\Gamma(\nu_h)} \left(\frac{2\sqrt{\nu_h}\,|h_h|}{\theta_h} \right)^{\nu_h} \mathcal{K}_{\nu_h} \left(\frac{2\sqrt{\nu_h}\,|h_h|}{\theta_h} \right) \tag{2.5}$$

Equation (2.4) is the anisotropic form of the n-dimensional Matérn covariance function with hyperparameters $\psi = (\nu_1, \ldots, \nu_n, \theta_0, \ldots, \theta_n) \in \mathbb{R}_+^{2n+1}$. The Matérn covariance function allows tuning of the mean square differentiability of $y(\cdot)$, which is not possible with the power exponential family. Let $\lceil \nu \rceil$ denote the ceiling of ν. Then functions drawn from a GP having Matérn covariance have a.s. continuously differentiable paths of order $\lceil \nu \rceil - 1$. This may result in a better prediction for nonsmooth functions.

In GP modeling, prior knowledge about the unknown function $y(\cdot)$ is represented by a GP with parameterized mean and covariance functions, i.e., the sample set $\mathcal{S}$ has a known multivariate normal distribution with unknown hyperparameters. The prior is an assumption and does not make use of $\mathcal{S}$. To use the prior to predict $y(\cdot)$ at general points, it must be updated using $\mathcal{S}$. Let $y_0 \triangleq y(x_0)$ denote the unknown function value to be predicted at a point x_0 and let $y^{(k)} = [y(x^{(1)}), \ldots, y(x^{(k)})]^T$ be the vector of observations[1]. By assumption, $\mathcal{S} \cup \{y_0\}$ has a joint normal distribution

[1] The observations $y^{(k)}$ are considered values of a *single* random function drawn from the prior $Y(\cdot, \Omega)$, i.e., $[y(x^{(1)}), \ldots, y(x^{(k)})]^T = [Y(x^{(1)}, \omega), \ldots, Y(x^{(k)}, \omega)]^T$. In the fictional setup of describing the deterministic function $y(\cdot, \omega)$ as a sample path of $Y(\cdot, \Omega)$, the deterministic observations $y^{(k)}$ as well as y_0 are also random variables, but in this chapter they are written using the lower case y to remind the reader that they are not true random variables.

with specified covariance structure, i.e.,

$$\begin{pmatrix} y_0 \\ y^{(k)} \end{pmatrix} \sim \mathcal{N}_{1+k} \left[\begin{pmatrix} \mathbf{f}_0^T \\ \mathbf{F} \end{pmatrix} \beta, \begin{pmatrix} \Bbbk(x_0, x_0) & \mathbf{k}_0^T \\ \mathbf{k}_0 & \mathbf{K} \end{pmatrix} \right] \tag{2.6}$$

where $\mathbf{f}_0 = \mathbf{f}(x_0) = [f_1(x_0), \ldots, f_p(x_0)]^T$ is the $p \times 1$ vector of regressors at x_0, $\mathbf{F}$ is the $k \times p$ matrix of regressors having (i,j)th element $f_j(x^{(i)})$ for $i = 1, \ldots, k$, $j = 1, \ldots, p$, β is the $p \times 1$ vector of unknown regression coefficients, $\mathbf{k}_0 = \left[\Bbbk(x_0, x^{(1)}), \ldots, \Bbbk(x_0, x^{(k)})\right]^T$ is the $k \times 1$ covariance vector, and the $k \times k$ matrix $\mathbf{K} = \Bbbk(x^{(i)}, x^{(j)})$ for $i, j = 1, \ldots, k$ is a covariance matrix of known structure but with unknown parameters β and θ to be determined from the sample set. It will be shown later that it is useful to consider the unknown parameters from a Bayesian point of view.

The simplest and most common Bayesian approach used to obtain the posterior predictive distribution $p(y_0|y^{(k)})$ is to assign the improper uniform prior to the mean β, i.e., $\beta \sim 1$, and to assume that the covariance hyperparameters are known, the values of which can be obtained later using some other method. The distribution (2.6) is then conditioned on β:

$$p(y_0, y^{(k)}|\beta) \sim \mathcal{N}_{1+k} \left[\begin{pmatrix} \mathbf{f}_0^T \\ \mathbf{F} \end{pmatrix} \beta, \begin{pmatrix} \Bbbk(x_0, x_0) & \mathbf{k}_0^T \\ \mathbf{k}_0 & \mathbf{K} \end{pmatrix} \right] \tag{2.7}$$

The posterior predictive can be obtained from

$$\begin{aligned} p(y_0|y^{(k)}) &= \int_\beta p(y_0, \beta|y^{(k)}) d\beta \\ &= \int_\beta p(y_0|y^{(k)}, \beta) p(\beta|y^{(k)}) d\beta \end{aligned} \tag{2.8}$$

The first term in the integrand of (2.8) is obtained from conditioning arguments [104, lemma B.1.2]:

$$p(y_0|y^{(k)}, \beta) \sim \mathcal{N}\left(\mu(x), \sigma_0^2(x)\right) \tag{2.9}$$

where

$$\mu(x) = \mathbf{f}_0^T \beta + \mathbf{k}_0^T \mathbf{K}^{-1}(y^{(k)} - \mathbf{F}\beta) \tag{2.10a}$$

$$\sigma_0^2(x) = \mathbb{k}(x_0, x_0) - \mathbf{k}_0^T \mathbf{K}^{-1}\mathbf{k}_0 \tag{2.10b}$$

Often, equations (2.10) will be reported in the literature as the mean and variance of the posterior predictive, but these expressions do not marginalize the uncertainty in β. Figure 9 illustrates the conditioning process based on equations (2.10). Figure 9a shows four sample functions drawn from a zero mean, unit variance GP prior and Figure 9b shows four sample functions (dashed lines) drawn from the posterior after two observations. The mean function (solid line, equation (2.10a)), represents the mean of all posterior sample functions, i.e., the mean of all functions conditioned on the observations, and is a function of x. In both plots the shaded area represents an uncertainty region equal to twice the standard deviation of the process. This region represents other functions that pass through the observations. Far from the observations, the mean and variance return to 0 and ±2, respectively, which are the mean and variance of the GP prior.

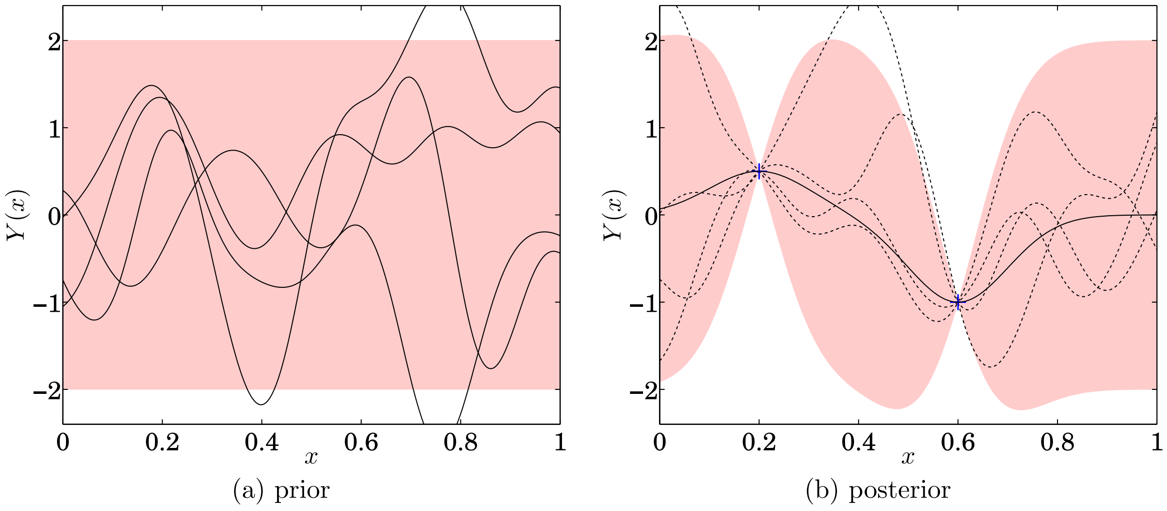
The second term in the integrand of (2.8) is obtained from Bayes' rule:

$$\begin{aligned} p(\beta|y^{(k)}) &= \frac{p(y^{(k)}|\beta)p(\beta)}{\int_\beta p(y^{(k)}|\beta)p(\beta)d\beta} \\ &= \mathcal{N}_k\left(\mathbf{F}\beta, \mathbf{K}\right) \cdot 1 \\ &\propto \exp\left[-\frac{1}{2}\beta^T \mathbf{F}^T \mathbf{K}^{-1}\mathbf{F}\beta + (\mathbf{F}^T\mathbf{K}^{-1}y^{(k)})^T\right] \\ &\sim \mathcal{N}_k(\hat{\beta}, \mathbf{\Sigma}) \end{aligned} \tag{2.11}$$

by [104, lemma B.1.1], where

$$\hat{\beta} = \left(\mathbf{F}^T\mathbf{K}^{-1}\mathbf{F}\right)^{-1}\left(\mathbf{F}^T\mathbf{K}^{-1}y^{(k)}\right) \tag{2.12a}$$

$$\mathbf{\Sigma} = \left(\mathbf{F}^T\mathbf{K}^{-1}\mathbf{F}\right)^{-1} \tag{2.12b}$$



(a) prior (b) posterior

Figure 9: Illustration of the GP conditioning process; observations shown as crosses. Shaded area represents the uncertainty region equal to twice the standard deviation of the respective distribution. Adapted from [94].

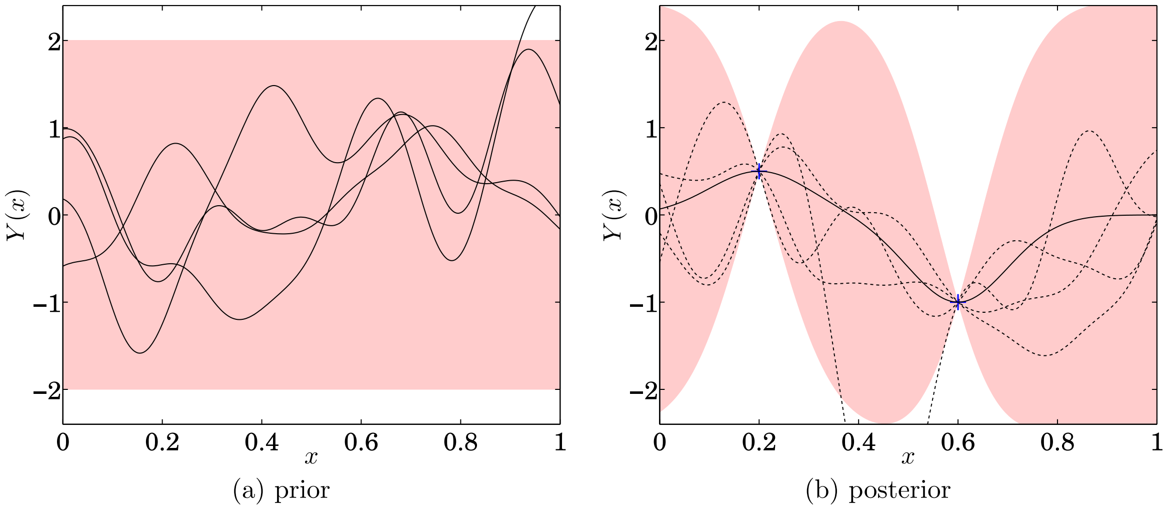
Substituting equations (2.9) – (2.12) into (2.8) and carrying out the integration results in the following expressions for the mean and variance of the posterior predictive:

$$
\begin{aligned}
\hat{Y}(x) &= \mathbb{E}[y_0 | y^{(k)}] \\
&= \mathbf{f}_0^T \hat{\beta} + \mathbf{k}_0^T \mathbf{K}^{-1} (y^{(k)} - \mathbf{F}\hat{\beta}) && \text{(2.13a)} \\
s^2(x) &= \mathbb{E}\left[(y_0 - \hat{Y}(x))^2 \right] \\
&= \sigma_0^2 + \left(\mathbf{f}_0 - \mathbf{F}^T \mathbf{K}^{-1} \mathbf{k}_0 \right)^T \mathbf{\Sigma} \left(\mathbf{f}_0 - \mathbf{F}^T \mathbf{K}^{-1} \mathbf{k}_0 \right) && \text{(2.13b)}
\end{aligned}
$$

Figure 10 illustrates the conditioning process based on equations (2.13). Compared with Figure 9, the uncertainty region of the posterior process has increased to account for the uncertainty in β.

2.2.2 Training The GP Model

The sample functions shown in the previous section were random draws from *known* GP priors with known hyperparameters. The core problem of GP modeling is how to



(a) prior

(b) posterior

Figure 10: Illustration of the GP conditioning process with β marginalized; observations shown as crosses. Shaded area represents the uncertainty region equal to twice the standard deviation of the respective distribution.

determine the hyperparameters $\theta = (\theta_0, \dots, \theta_n)$ [2] to best model a set of observations. In a fully Bayesian approach, priors $p(\theta)$ are assigned to each of the hyperparameters which are then marginalized into the posterior distribution using Bayes' rule

$$p(y_0|y^{(k)}) = \frac{\int_\theta p(y_0|y^{(k)},\theta)p(y^{(k)}|\theta)p(\theta)d\theta}{\int_\theta p(y^{(k)}|\theta)p(\theta)d\theta} \tag{2.14}$$

Equation (2.14) is generally nonanalytic, thus some numerical method must be used to compute the integrals, e.g., Bayesian Monte Carlo [93]. A novel method for training a fully Bayesian GP model will be described shortly. A common approximation is obtained via maximum likelihood estimation (MLE), which assumes that $p(y^{(k)}|\theta)$ is degenerate with all its mass located at $\theta = \hat{\theta}_{\text{MLE}}$, i.e., $p(y^{(k)}|\theta) \sim \delta(\theta - \hat{\theta}_{\text{MLE}})$. MLE is used extensively [104, 56, 94] due to its simplicity. The problem of estimating the hyperparameters then becomes

$$\begin{aligned} &\underset{\theta\in\mathbb{R}^{n+1}}{\text{maximize}} \quad \log \ell(\theta|y^{(k)}) \\ &\text{subject to} \quad \theta > 0 \end{aligned} \tag{2.15}$$

[2] This set is for a zero mean GP prior; for a constant mean prior, the set of hyperparameters includes β and also $\sigma^2 \triangleq \theta_0$ is sometimes treated separately.

where $\ell(\theta|y^{(k)}) \triangleq p(y^{(k)}|\theta)$ is a likelihood function. Under a multivariate normal assumption and general covariance structure, the likelihood function is

$$\ell(\theta|y^{(k)}) = (2\pi)^{-n/2} \left|\mathbf{K}\right|^{-1/2} \exp\left[-\frac{1}{2}(y^{(k)} - \mathbf{F}\beta)^T \mathbf{K}^{-1}(y^{(k)} - \mathbf{F}\beta)\right] \tag{2.16}$$

where $|\cdot|$ denotes the determinant of a matrix. The log-likelihood function is then

$$\log \ell(\theta|y^{(k)}) = -\frac{1}{2}\left[k \log 2\pi + \log|\mathbf{K}| + (y^{(k)} - \mathbf{F}\beta)^T \mathbf{K}^{-1}(y^{(k)} - \mathbf{F}\beta)\right] \tag{2.17}$$

Problem (2.15) is then solved to determine the hyperparameters θ. Problem (2.15) is a bound constrained nonlinear programming problem for which there are several applicable techniques. To solve this problem, either a stochastic method can be used to find a good starting point for a local minimizer or a multistart method can be used with a gradient-based solver. A closed-form equation for the gradient of the log-likelihood function, equation (2.17), may be useful [94]:

$$\begin{aligned}
\frac{\partial \log \ell(\theta|y^{(k)})}{\partial \theta_j} &= -\frac{1}{2}\left[\mathrm{Tr}\left(\mathbf{K}^{-1}\frac{\partial \mathbf{K}}{\partial \theta_j}\right) - (y^{(k)} - \mathbf{F}\beta)^T \mathbf{K}^{-1}\frac{d\mathbf{K}}{d\theta_j}\mathbf{K}^{-1}(y^{(k)} - \mathbf{F}\beta)\right] &\quad (2.18)\\
&= \frac{1}{2}\mathrm{Tr}\left(\left(\alpha\alpha^T - \mathbf{K}^{-1}\right)\frac{\partial \mathbf{K}}{\partial \theta_j}\right) \quad \text{where} \quad \alpha = \mathbf{K}^{-1}y^{(k)} &\quad (2.19)
\end{aligned}$$

If $\mathcal{S}$ is corrupted by output-dependent noise such that the observed values are now $z(x) = y(x) + \epsilon$, a noise term $\epsilon_n = \delta_{ii}\theta_n$ can be appended to the covariance function, where δ_{ii} is the Kronecker delta and θ_n is the noise variance which is treated as an additional hyperparameter. The GP will no longer interpolate the observations; rather, it will regress them. See §5.4 for details.

2.2.3 Limitations & Scalability

The computational effort required to train a GP model with anisotropic covariance function is $\mathcal{O}(n^3)$. Preliminary results indicate that training GP models with anisotropic covariance functions works well for small n, say, $n \leq 3$, but may become computationally intractable for larger n. One way remedy the curse of dimensionality is to use an isotropic covariance function such that $\theta_1 = \theta_2 = \cdots = \theta_n$. This forces the GP model to employ the same lengthscale parameter θ in every coordinate direction, which may reduce modeling flexibility in some cases, but this is not guaranteed. Examples of two-dimensional random functions drawn from GPs having anisotropic and isotropic SE covariance functions are shown in Figure 11. The utilization of an isotropic covariance function is crucial to the success of the novel model fitting method that is described in §2.2.5, since this choice of covariance functions allows the method to bypass the curse of dimensionality at the model fit step (and also at the ISC maximization step).

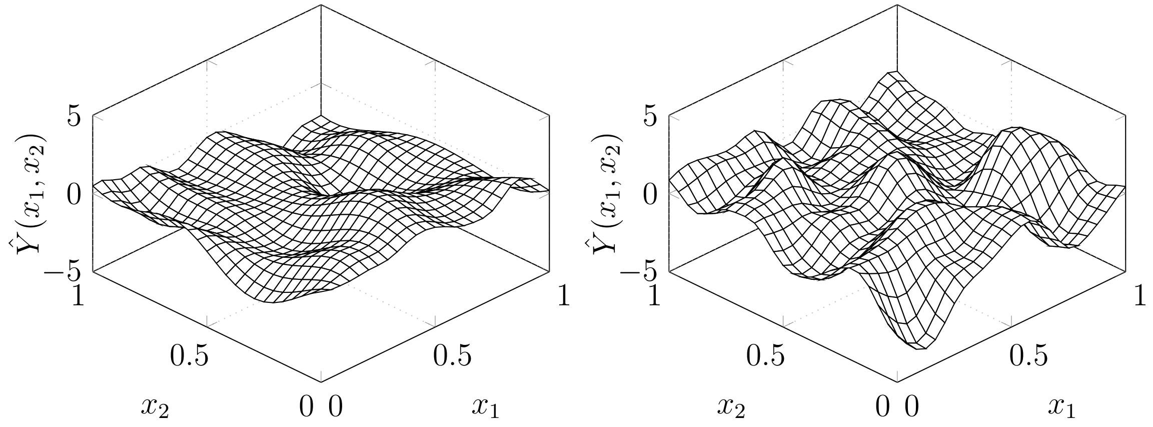


Figure 11: Illustration of anisotropic and isotropic SE covariance functions. Left, $\theta_1 = 0.1$, $\theta_2 = 0.2$; right, $\theta_1 = \theta_2 = 0.1$

2.2.4 Inadequacy Of Likelihood-Based Approaches

Conceptually, MLE leads to the values of the hyperparameters that were most likely to have generated $\mathcal{S}$. In general, the more observations, the more peaked $\ell(\theta|y^{(k)})$ becomes, and MLE is justified (see Figure 12). When data is scarce, e.g., in the case of expensive problems, there is more mass away from the peak of $\ell(\theta|y^{(k)})$ and the entire distribution becomes important [94]. In this case the fully Bayesian approach must be used to marginalize the uncertainty in θ.

In general, $\ell(\theta|y^{(k)})$ is multimodal with every maximum corresponding to a particular interpretation of the data. In Figure 13, an example with two local maxima is shown together with the corresponding posterior predictive distributions. The global maximum corresponds to a relatively complicated model with low noise, whereas the nonglobal maximum corresponds to a simpler model with a higher noise level. Neither of the two models in Figure 13 is correct, since the data were generated from a zero mean, unit variance GP with SE correlation function having[3] $\theta = 1$ and noise variance $\theta_n = 0.1$. George P. Box famously noted that "essentially, all models are wrong, but some are useful" [19]; a human observer may find it difficult to decide which of the two models, if any, is correct, or even which is the most probable. The true hyperparameters are represented by the diamond marker on the log-likelihood plot and the corresponding true model is shown in the bottom left of Figure 13. The true model is a compromise between the two models predicted by MLE. This phenomenon arises from the interactions between the terms in the log-likelihood function, equation (2.17). The second term inside the brackets in equation (2.17) involves the determinant of the covariance matrix, which decreases monotonically with increasing lengthscale factor θ, and thus favors less complex models. The last term is a data-fit term which increases monotonically with increasing lengthscale. For sparse data, these competing effects can lead to unreasonable models. As more data is added, the complexity penalty $\log |\mathbf{K}|$ becomes more severe, discouraging short lengthscales. According to the Bayesian methodology, one ought to marginalize the hyperparameters to obtain a posterior predictive that integrates the effects of many lengthscales.

Besides MLE, there are other approaches for computing estimates to the hyperparameters θ [104, §3.3.2], [94, §2.1.1]. These include the maximum *a posteriori* (MAP) approach, restricted MLE (or marginal MLE [94]), and cross-validation. In the literature, MLE appears to be the most popular method for training a GP. The issue of which method is best has not been studied in depth [104, §3.3.2]. However, these alternative methods suffer from similar drawbacks as MLE since each of these methods

[3] In the case $n = 1$, the subscript h in θ_h may be omitted and it is understood that $\theta = \theta_1$

are likelihood-based approaches.

2.2.5 Fully Bayesian Approach

In a fully Bayesian setting, all unknown parameters of the predictive distribution $p(Y|y^{(k)})$ are assigned prior distributions. While Bayes' rule describes how to update a prior after making observations, there is no law of probability that describes what the prior should be. The general problem of selecting priors is unresolved and the necessity of having to guess a prior is the source of opposition of frequentists to the Bayesian methodology [6].

The most fundamental intuition one can appeal to when assigning priors is the well-known *principle of indifference*, or *principle of insufficient reason*. The principle of indifference asserts that in the absence of positive ground for assigning unequal probabilities to a set of arguments, equal probabilities must be assigned to each of them [58], i.e., to say that the probabilities are equal is the precise way of saying that there exists no ground for choosing between the alternatives [51].

An important class of priors is the *conjugate* prior. A prior is conjugate to its posterior if the posterior and prior are in the same family of distributions. Conjugate priors lead to closed-form posteriors, but may not always be reasonable choices. The number of conjugate priors for a family of likelihood functions is very limited [34]. Another important class is the *improper* prior, which offers a simple solution to the marginalization problem, but may not always lead to a proper posterior. Improper priors do not satisfy $\int p(x)dx = 1$ and are also *uninformative*, i.e., they represent vague or general information about a random variable. The reason why some improper priors such as the uniform improper prior lead to proper posteriors is that these priors are the probability limits of proper priors. For example, computing a GP posterior with a normal prior on the mean and then taking the limit as the variance of the mean goes to infinity gives the same result as a GP posterior with a uniform improper prior

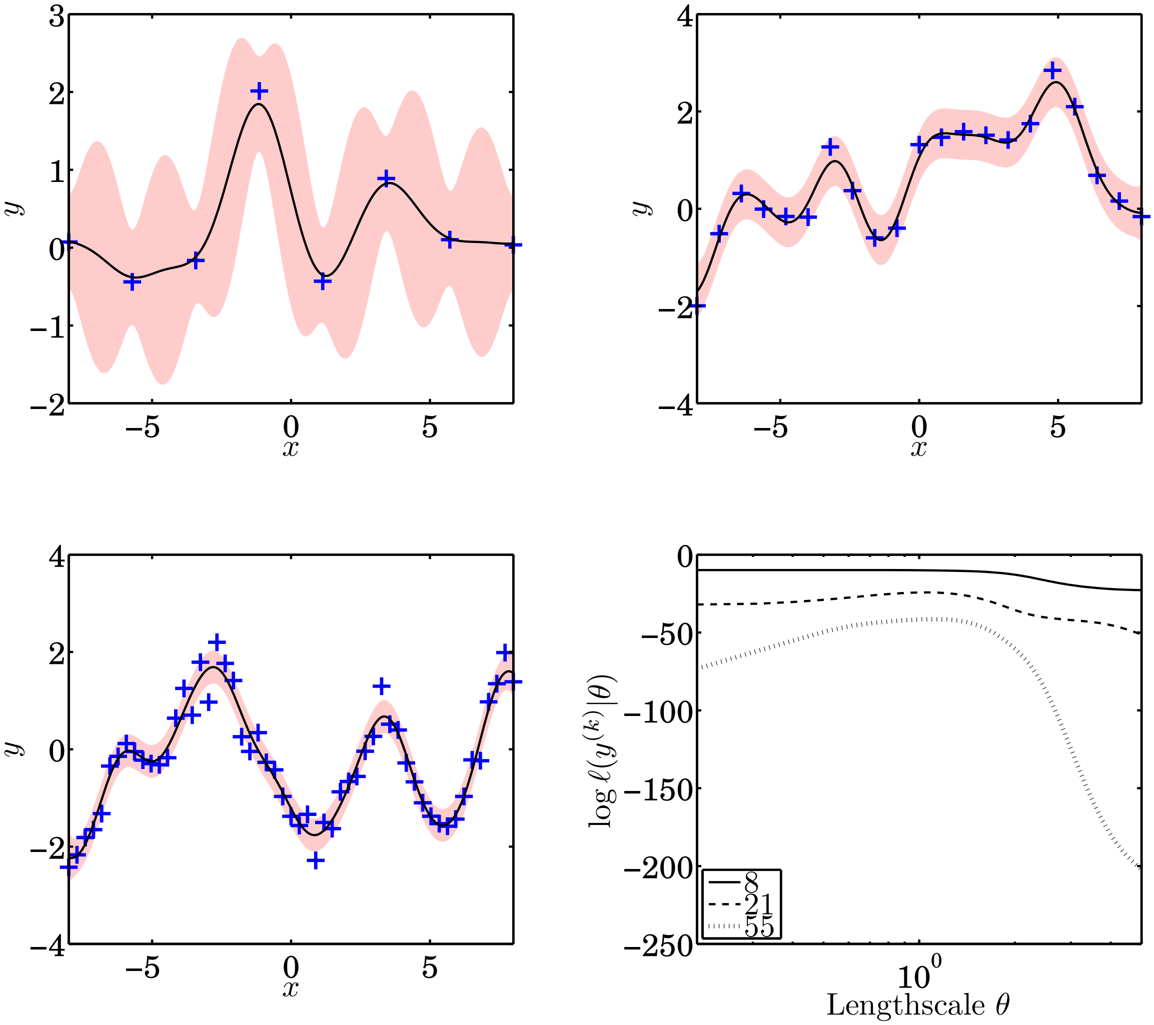


Figure 12: The effect of the number of observations on the likelihood. Observations (crosses) drawn from a zero mean, unit variance GP with SE covariance function having $\theta = 1$ and noise variance $\theta_n = 0.1$. GP models were conditioned to the observations using the *a priori* parameters and the log-likelihood functions $\log \ell(\theta|y^{(k)})$ are shown as a function of θ. Shaded area denotes an uncertainty region of $\pm 2s$.

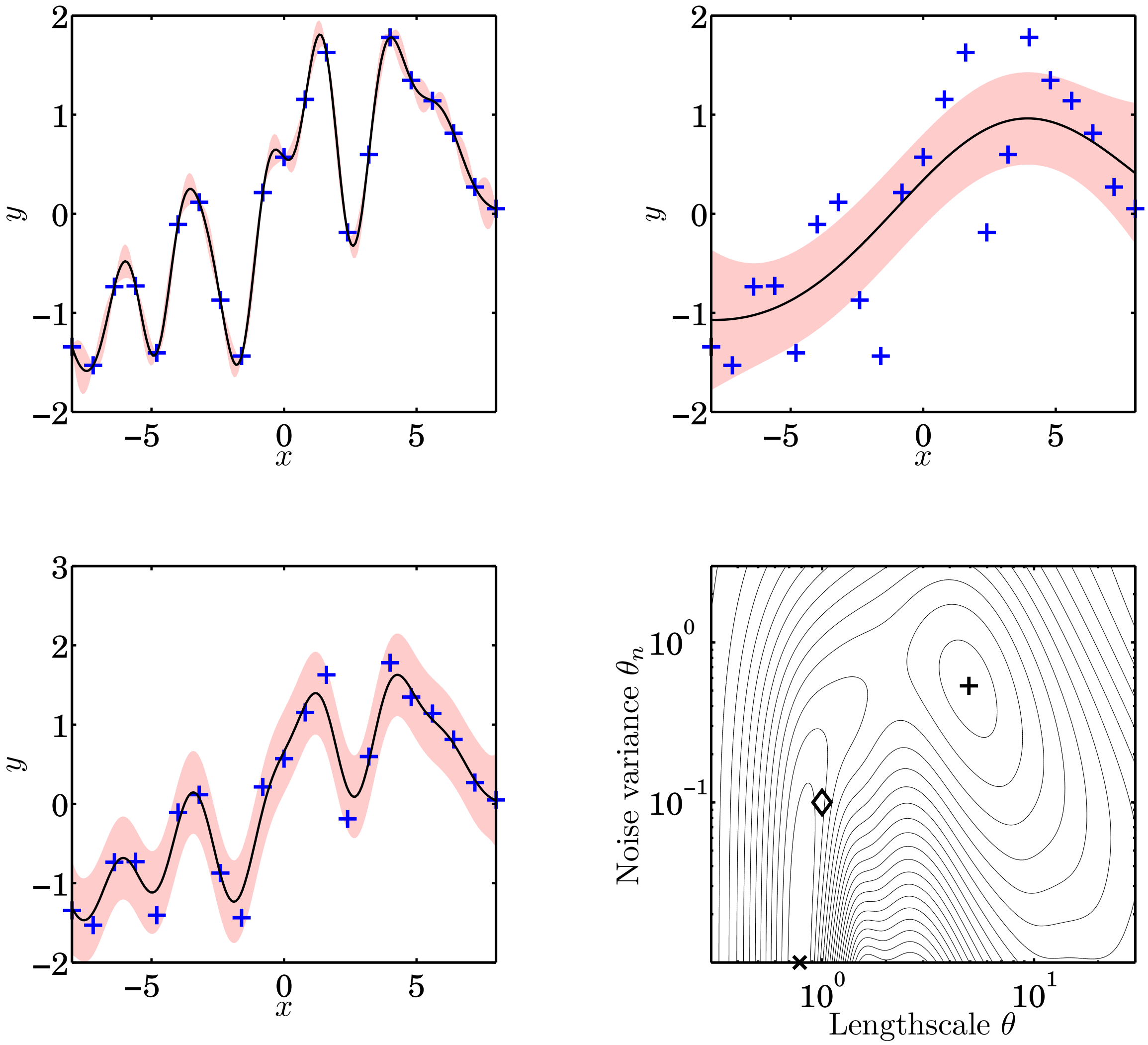


Figure 13: Log-likelihood as a function of the hyperparameters θ and θ_n, bottom right; MLE fit corresponding to the global maximum log-likelihood (× symbol on likelihood plot), top left; MLE fit corresponding to the local maximum likelihood (+, top right; function corresponding to the true hyperparameters $\theta = 1$ and $\theta_n = 0.1$ that also generated the data ($\diamond$ symbol on log-likelihood plot), bottom left. Shaded area denotes an uncertainty region of $\pm 2s$.

on the mean.

Consider the case of a Gaussian process $Y(\cdot)$ with unknown scalar constant mean β and covariance function $\Bbbk(\mathbf{h}) = \sigma^2 r(\mathbf{h})$. Assume that β and σ^2 are independent and that $\beta \sim 1$, i.e., β is assigned the uniform improper prior, and $\sigma^2 \sim IG(a_0, b_0)$, i.e., σ^2 follows an inverse Gamma distribution with shape parameter a_0 and scale parameter b_0.

A constant mean prior $\mathcal{N}(\beta, \sigma^2\mathbf{R})$ is placed on the unknown function $y(\cdot)$. The first goal is to obtain an expression for the predictive distribution $p(Y|y^{(k)}, \theta)$. Begin by writing

$$p(Y|y^{(k)}, \theta) = \int_{\sigma^2} p(Y|y^{(k)}, \sigma^2, \theta)p(\sigma^2|y^{(k)}, \theta)d\sigma^2 \tag{2.20}$$

The term $p(Y|y^{(k)}, \sigma^2, \theta)$ can be shown [104] to be distributed as $\mathcal{N}(\mu, \sigma^2\kappa^2)$ with mean and variance defined by equations (2.13), restated here to reflect the constant mean GP prior:

$$\begin{aligned}
\mu(x) &= \hat{\beta} + \mathbf{r}_0^T\mathbf{R}^{-1}(y^{(k)} - \hat{\beta}) \\
\hat{\beta} &= \frac{1^T\mathbf{R}^{-1}y^{(k)}}{1^T\mathbf{R}^{-1}1} \\
\kappa^2(x) &= 1 - \mathbf{r}_0^T\mathbf{R}^{-1}\mathbf{r}_0 + \frac{\left(1 - 1^T\mathbf{R}^{-1}\mathbf{r}_0\right)^2}{1^T\mathbf{R}^{-1}1}
\end{aligned}$$

The term $p(\sigma^2|y^{(k)}, \theta)$ in equation (2.20) can be computed from the prior $p(\sigma^2)$ and Bayes' rule:

$$p(\sigma^2|y^{(k)}, \theta) = \frac{p(y^{(k)}|\sigma^2, \theta)p(\sigma^2)}{\int_{\sigma^2} p(y^{(k)}|\sigma^2, \theta)p(\sigma^2)d\sigma^2} \tag{2.21}$$

The term $p(y^{(k)}|\sigma^2, \theta)$ in equation (2.21) can be computed from

$$\begin{aligned}
p(y^{(k)}|\sigma^2, \theta) &= \int_\beta p(y^{(k)}|\beta, \sigma^2, \theta)p(\beta)d\beta \\
&= \int_\beta \mathcal{N}(\beta, \sigma^2\mathbf{R}) \cdot 1 \cdot d\beta \\
&\propto (\sigma^2)^{-(k-1)/2} \exp\left[-\frac{1}{2\sigma^2}\left((y^{(k)})^T\mathbf{R}^{-1}y^{(k)}\right.\right. && (2.22) \\
&\qquad \left.\left. -(1^T\mathbf{R}^{-1}y^{(k)})^T\hat{\beta}\right)\right] && (2.23)
\end{aligned}$$

Since the terms in equation (2.21) are conjugate, carrying out the integration gives the analytical expression

$$p(\sigma^2|y^{(k)}, \theta) \sim IG(a_k, b_k) \tag{2.24}$$

with

$$a_k = a_0 + \frac{k-1}{2} \tag{2.25a}$$

$$b_k = b_0 + \frac{1}{2}\left[(y^{(k)})^T\mathbf{R}^{-1}y^{(k)} - \left(1^T\mathbf{R}^{-1}y^{(k)}\right)^T \hat{\beta}\right] \tag{2.25b}$$

That is, the conditional distribution of σ^2 given $y^{(k)}$ and θ is inverse Gamma with real shape and scale parameters a_k and b_k, respectively.

Substituting (2.24) into (2.20) and carrying out the integration yields

$$p(Y|y^{(k)}, \theta) \sim t_{\eta_k=2a_k}\left(\mu(x), \gamma_k^2(x) \triangleq (b_k/a_k)\kappa^2(x)\right) \tag{2.26}$$

That is, the conditional predictive distribution at x is a location-scale t distribution with $\eta_k = 2a_k$ degrees of freedom, location parameter $\mu(x)$, and scale parameter $\gamma_k^2(x) \triangleq (b_k/a_k)\kappa^2(x)$. The variance of this distribution is

$$\mathrm{Var}(Y|y^{(k)}, \theta) = \gamma_k^2(x)\frac{\eta_k}{\eta_k - 2}, \qquad \eta_k > 2 \tag{2.27}$$

Thus, the sample size k and a_0 are related through

$$k \geq \lceil 3 - 2a_0 \rceil \tag{2.28}$$

where $\lceil \cdot \rceil$ is the ceiling operator. Condition (2.28) must be met at all iterations, including the initial design.

In general, equation (2.14) must be approximated through some numerical technique. This procedure can become expensive since the integral (2.14) must be approximated at many points. The calculations can be greatly simplified by assuming a discrete prior $p(\theta)$, in which case the integral (2.14) becomes a summation over the p.m.f. The procedure using a discrete prior $p(\theta)$ is as follows. The predictive distribution $p(Y|y^{(k)})$ can be computed from

$$\begin{aligned} p(Y|y^{(k)}) &= \int_\theta p(Y,\theta|y^{(k)})d\theta \\ &= \int_\theta p(Y|y^{(k)},\theta)p(\theta|y^{(k)})d\theta \\ &= \sum_i p(Y|y^{(k)},\theta_i)p(\theta_i|y^{(k)}) \end{aligned} \tag{2.29}$$

where the last equality in equation (2.29) comes from the fact that $p(\theta_i|y^{(k)})$ is discrete[4]. Thus, $p(Y|y^{(k)})$ is a finite mixture distribution. The term $p(\theta_i|y^{(k)})$ is computed using Bayes' rule:

$$p(\theta_i|y^{(k)}) = \frac{p(y^{(k)}|\theta_i)p(\theta_i)}{\sum_i p(y^{(k)}|\theta_i)p(\theta_i)} \tag{2.30}$$

The term $p(y^{(k)}|\theta_i)$ is computed as

$$\begin{aligned} p(y^{(k)}|\theta_i) &= \int_{\sigma^2} p(y^{(k)}|\sigma^2,\theta_i)p(\sigma^2)d\sigma^2 \\ &= \frac{|\mathbf{R}|^{-1/2}}{(1^T\mathbf{R}1)^{1/2}}\frac{b_0^{a_0}}{b_k^{a_k}}\frac{\Gamma(a_k)}{\Gamma(a_0)} \end{aligned} \tag{2.31}$$

where $\Gamma(\cdot)$ is the gamma function. It can be shown that the mean and variance of the finite mixture distribution $p(Y|y^{(k)})$ are:

$$\mu = \sum_i p(\theta_i|y^{(k)})\mu_i \tag{2.32}$$

$$\sigma^2 = \sum_i p(\theta_i|y^{(k)})\left(\mu_i^2+\sigma_i^2\right) - \mu^2 \tag{2.33}$$

where $\mu_i = \mathbb{E}[Y|y^{(k)},\theta_i]$ and $\sigma_i^2 = \text{var}(Y|y^{(k)},\theta_i)$.

The fully Bayesian method is compared with the MLE method in Figure 14, where models are constructed through four observations. For the fully Bayesian method, the prior $p(\theta)$ is the uniform discrete prior with one hundred elements uniformly spaced in the log space $-7 \le \log\theta \le \log(2)$. The fully Bayesian approach marginalizes the

[4] In this context, the index i denotes all combinations of $\theta_1,\ldots,\theta_n$ in the domain of $p(\theta)$

uncertainty in the hyperparameters in the GP model, thus increasing the uncertainty bands around the prediction. This corrects the major drawback in MLE-based approaches, which is that the hyperparameters obtained by MLE can give a deceptive picture of the unknown function. It remains to define a method for assigning the discrete prior.

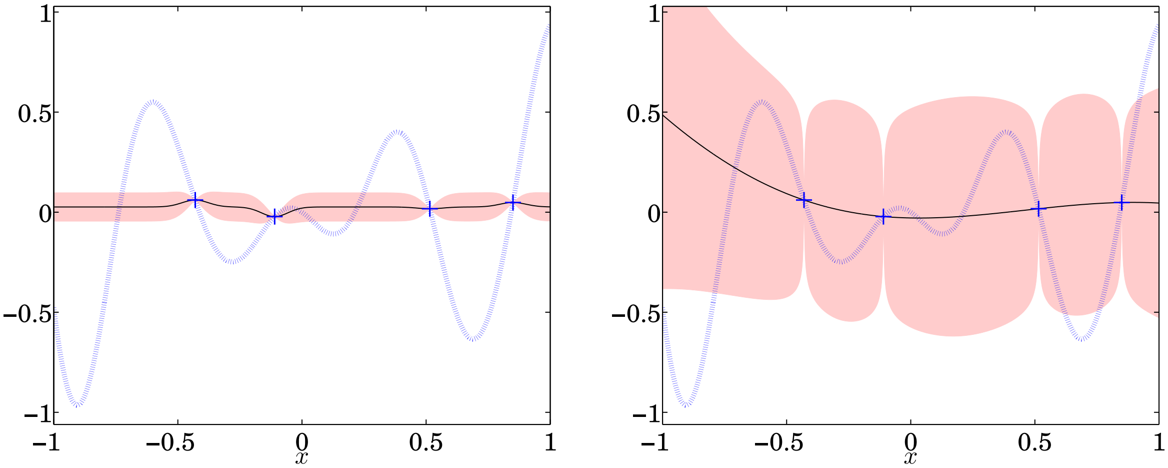


Figure 14: MLE fit (left) compared with fully Bayesian fit (right). Shaded area represents an uncertainty region of $\pm 2s$.

2.2.6 Contribution: Method For Assigning Hyperparameter Priors

The demonstration of the fully Bayesian approach in Figure 14 made use of a discrete prior, but little work is found in the literature describing how to assign a practical prior [79, 45]. In [79], a uniform prior over a fixed continuous domain is used for the hyperparameters, which is then marginalized into the predictive distribution using BMC (Bayesian Monte Carlo, see Appendix B). BMC employs MLE and thus reintroduces the associated drawbacks into the method, although it is not clear as to what degree they impact the performance. According to numerical results in Appendix B, BMC may produce a substantial number of large outliers for some problems. In addition, a truncated prior may be a poor assumption since the hyperparameter values excluded from the prior may be relevant, while those included may provide negligible contribution to the final model.

This section begins by demonstrating an emergent property of Bayesian inference which penalizes values of θ that result in unnecessarily complex models and overly simple models. In the context of GPs and the described covariance functions (2.2.1), complexity is measured by the value of θ; low θ values result in highly nonlinear models that are deemed complex. The penalization property is used as the basis of an iterative method that attempts to construct a prior $p(\theta)$ that results in a posterior $p(\theta|y^{(k)})$ that has most of its mass located on the interior of the distribution (as opposed to on the boundary). The combinations of θ that contribute the most mass to $p(\theta|y^{(k)})$ are then used to construct the fully Bayesian model for the unknown function.

The penalization mechanic is illustrated in Figure 15, where four different models $\hat{Y}(x) = \mathbb{E}[Y|y^{(k)}, \theta]$ are shown along with their convex combination, equation (2.29), which is the predictive distribution (dotted line). The posterior probabilities $p(\theta|y^{(k)})$ are computed for the hyperparameter prior

$$p(\theta) = \begin{cases} \frac{1}{4}, & \log_{10}\theta = -2, -1, 0, 1 \\ 0, & \text{else} \end{cases} \tag{2.34}$$

Based on the posterior values $p(\theta|y^{(k)})$, the hyperparameters $\log_{10}\theta = -2, -1$ make a negligible contribution to the predictive distribution, with $\theta = 1$ being preferred with a posterior probability of 0.93162. There is also the possibility that a model can be "too smooth," i.e., its θ value is so large that the model begins to approach the mean of the data. In theory, the mean function interpolates the data for all θ, but in practice, large values of θ can lead to poor conditioning; the solution to this is to introduce artificial noise into the model so that it regresses the data (see §5.4). These competing effects lead to the nonuniform posterior distribution for $p(\theta|y^{(k)})$ shown in Figure 16

In the example above, a user may wish to refine the prior 2.34 to include larger values of θ because for $\theta = 10$, the mass on the boundary is $p(\theta|y^{(k)}) = 0.051483$,

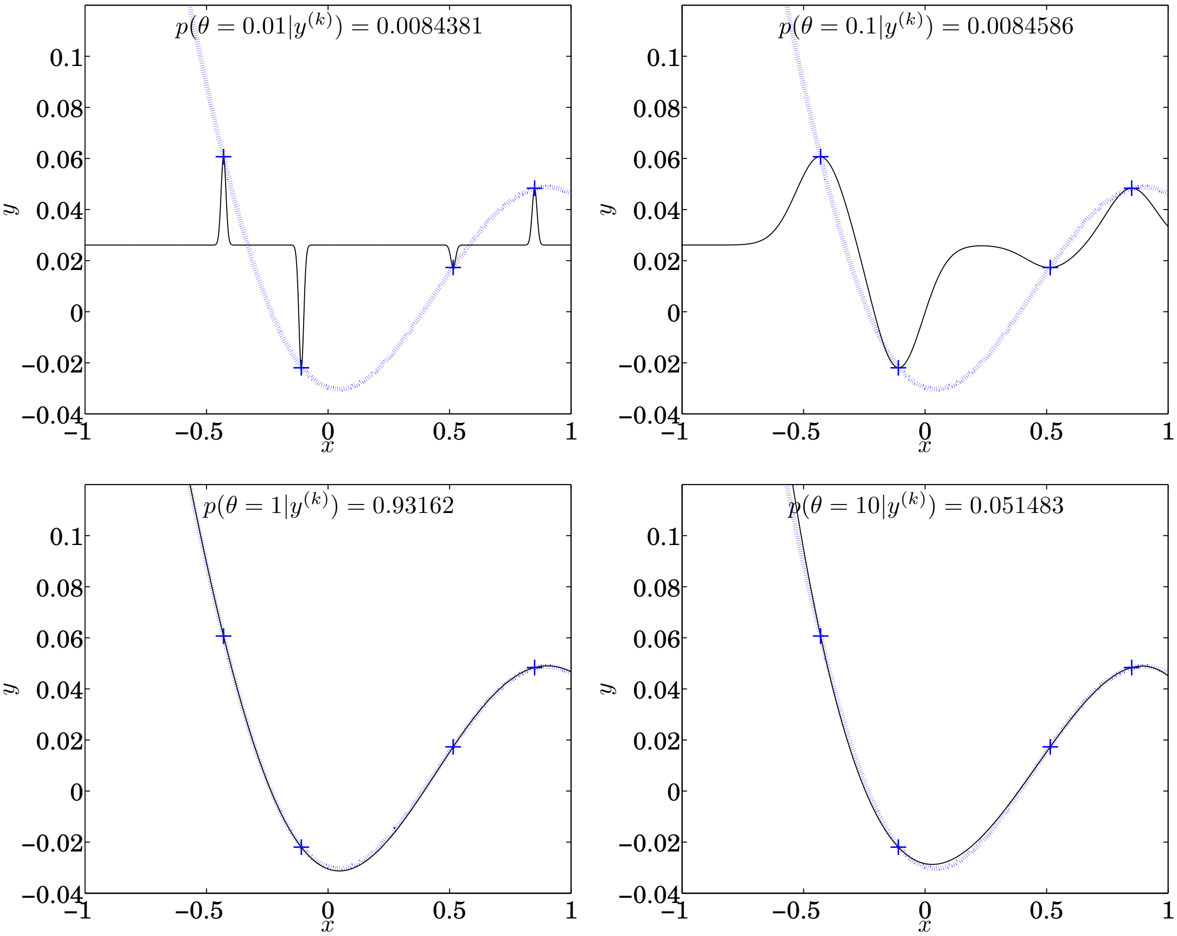


Figure 15: Bayesian penalization of unnecessarily complex and overly simple models. Conditional models (solid lines) and weighted sum model (dotted line).

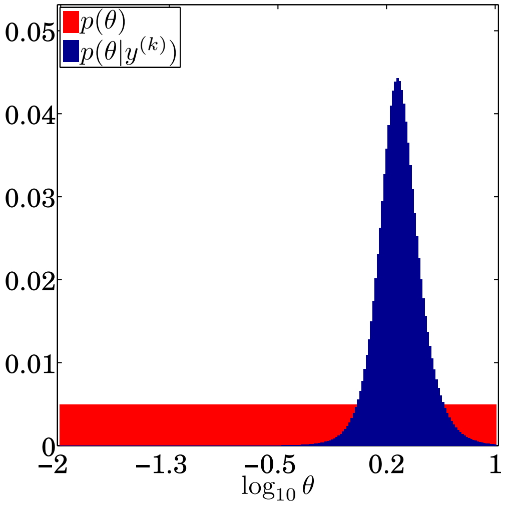


Figure 16: Dense posterior distribution for the problem in Figure 15.

indicating that some values $\theta > 10$ may provide some additional significant contribution to the model. An iterative strategy can be developed that generates θ such that most of the mass of $p(\theta|y^{(k)})$ occurs on the interior of the p.m.f. The strategy is outlined below.

1. Select a prior $p(\theta)$ and compute $p(\theta|y^{(k)})$.

2. (**Iteration step**) If any $p(\theta|y^{(k)})$ on the boundary of θ are large, extend θ in that direction by some amount and recompute $p(\theta|y^{(k)})$.

3. Construct the fully Bayesian model using the elements from θ corresponding to most of the mass in $p(\theta|y^{(k)})$.

The last step above reduces the analytical effort since for small $p(\theta|y^{(k)})$, the conditional model $p(y_0|y^{(k)}, \theta)$ in equation (2.29) is insignificant. The strategy is illustrated in Figures 18 – 20 for the test function shown in Figure 17, which is the $-\log(-y)$ transformation of the Shekel's Foxholes problem B-13. The full prior and posterior obtained by iteration is also shown in Figure 17. In general, the domain of θ that contains most of the mass is unknown since the function is unknown. Figure 18 shows $p(\theta|y^{(k)})$ based on an initial prior $p(\theta)$, which is an assumption, and the resulting GP model for the test function. The posterior probabilities are nearly uniform. This can mean one of two things. The first is that the initial prior was chosen close to the "true" prior and each mode is favored equally. The second is that all elements of the prior are incorrect and unfavorable, i.e., the available values of θ were insufficient to infer anything useful about the underlying function. The latter possibility is far more likely to occur than the former and this claim is validated based on the right-hand panel in Figure 18, which is the Bayesian model constructed from the prior. A third interpretation of this result is that the iterative procedure is a process to "uncover" the true prior.

After one iteration, the prior $p(\theta)$ is expanded to include longer lengthscales, resulting in the posterior distribution shown in Figure 19. The posterior is starting to show a distribution but there is still a large amount of mass on the boundary of θ in the positive log direction. The resulting GP model, constructed from the prior, is starting to model the major features of the true function. At this point, a user can terminate the method, but in general the true function is unknown and this sort of comparison cannot be made.

A third iteration results in the further expansion of $p(\theta)$, which gives the posterior shown in Figure 20. Most of the mass is contained on the interior and there is little mass on the boundary, thus this is the last iteration. The resulting model constructed from the prior (Figure 20) now shows many subtle features of the true function. The top six favored modes, normalized to $\hat{Y}(x) \in [0, 1]$ for comparison, are shown in Figure 21. The sixth mode is approximately equal to the value of θ found by MLE. Compared with MLE (Figure 22), the fully Bayesian approach is better able to model nonlinear functions that require combinations of hyperparameters to capture the landscape features. It is envisioned that, when using this method to construct the model in Figure 20, a search algorithm would be able to accurately locate the global minimum of the true function with only a few more function evaluations.

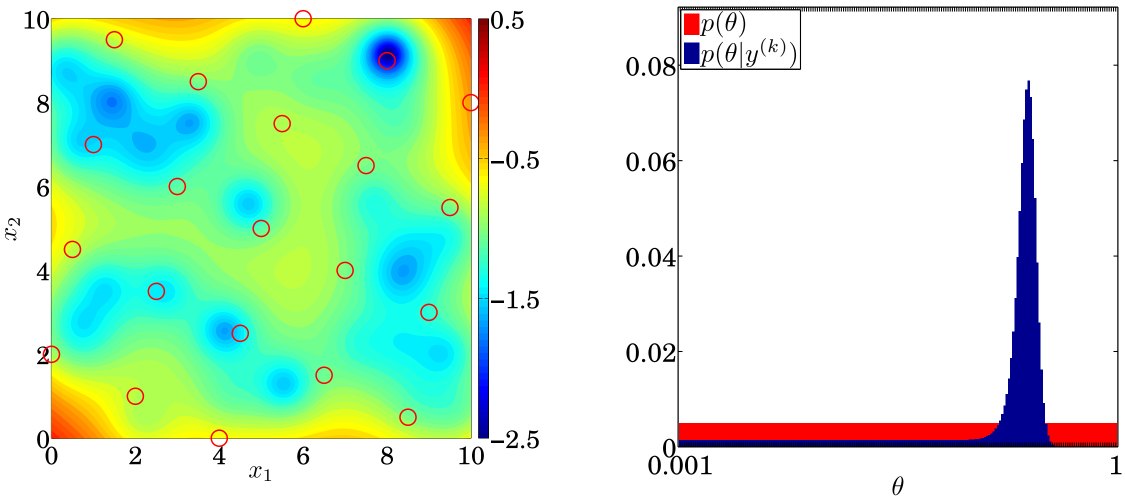
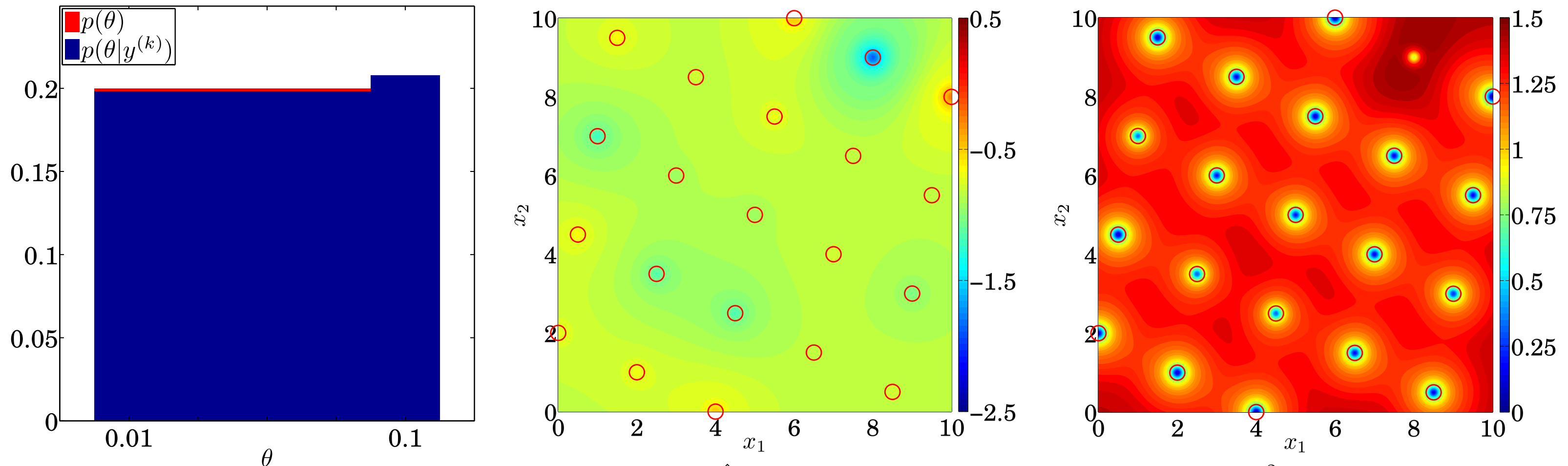


Figure 17: Test function used to demonstrate the proposed method along with dense hyperparameter posterior.

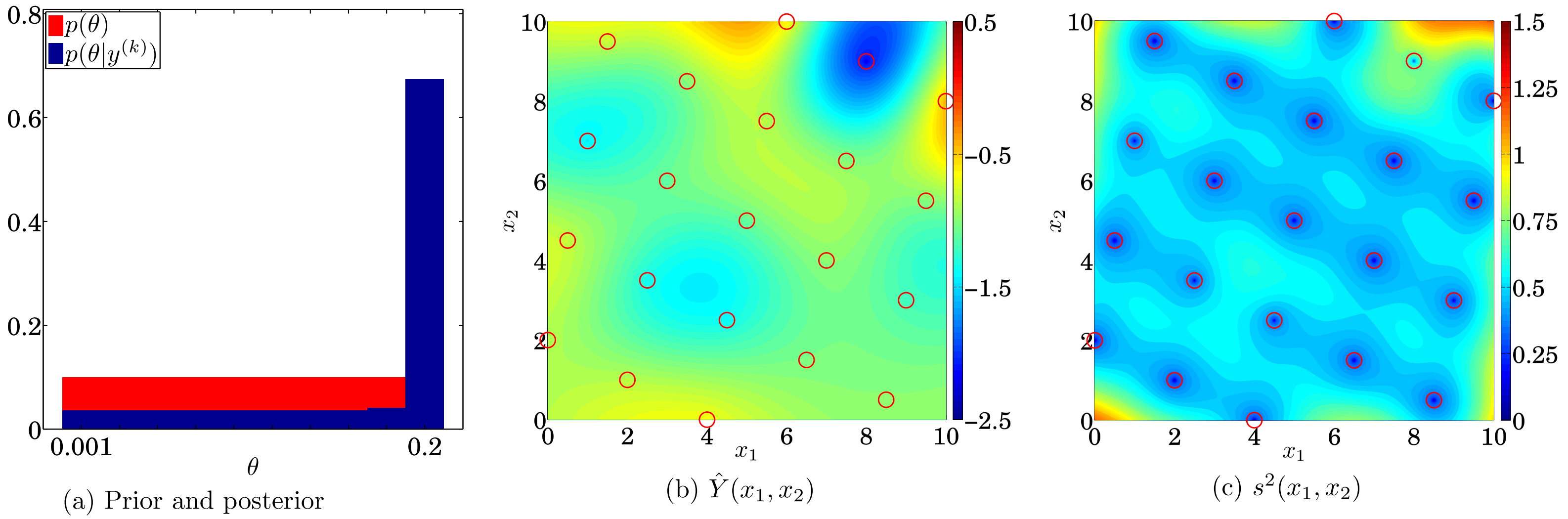


(a) Prior and posterior

(b) $\hat{Y}(x_1, x_2)$

(c) $s^2(x_1, x_2)$

Figure 18: Fully Bayesian model of the test function based on a uniform p.m.f. $p(\theta)$ (left) shown as a bar plot. The posterior probabilities $p(\theta|y^{(k)})$ are nearly uniform, indicating that they are all useless. The available lengthscales are in fact too short, as shown in GP model (middle) and posterior variance (right).

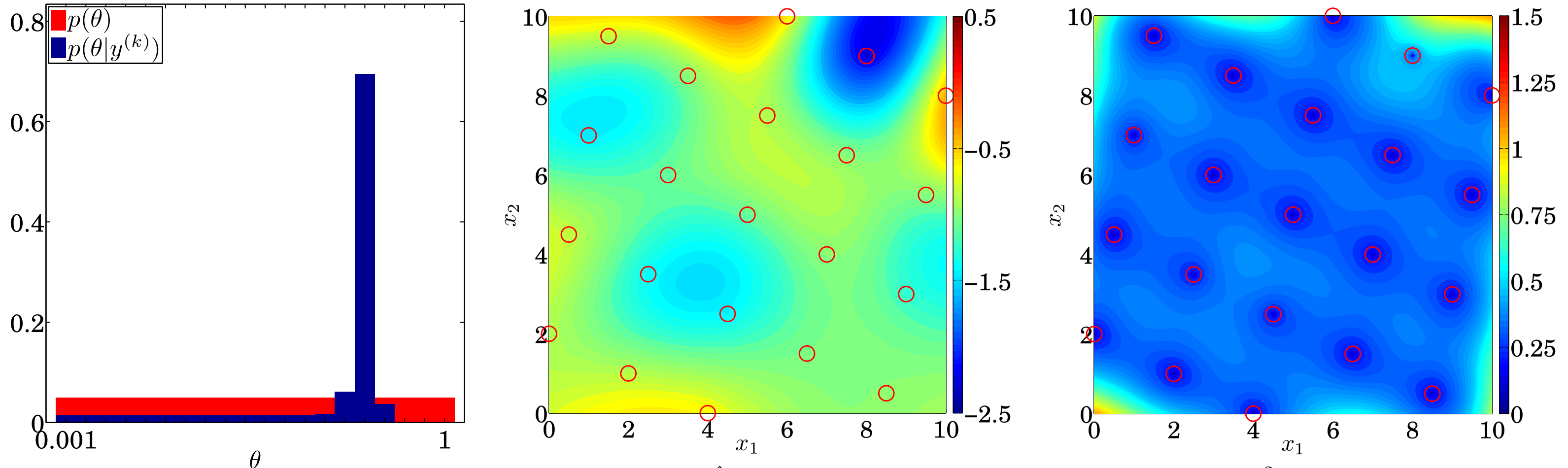


(a) Prior and posterior

(b) $\hat{Y}(x_1, x_2)$

(c) $s^2(x_1, x_2)$

Figure 19: Fully Bayesian model of the test function based on revised uniform p.m.f. $p(\theta)$ (left). The domain of the p.m.f. has been expanded. A portion of the full posterior distribution is revealed, but the large amount of mass on the boundary indicates that the prior needs to be further expanded. The prior is complete enough to model the major features of the true function, but this comparison is not possible for unknown functions.

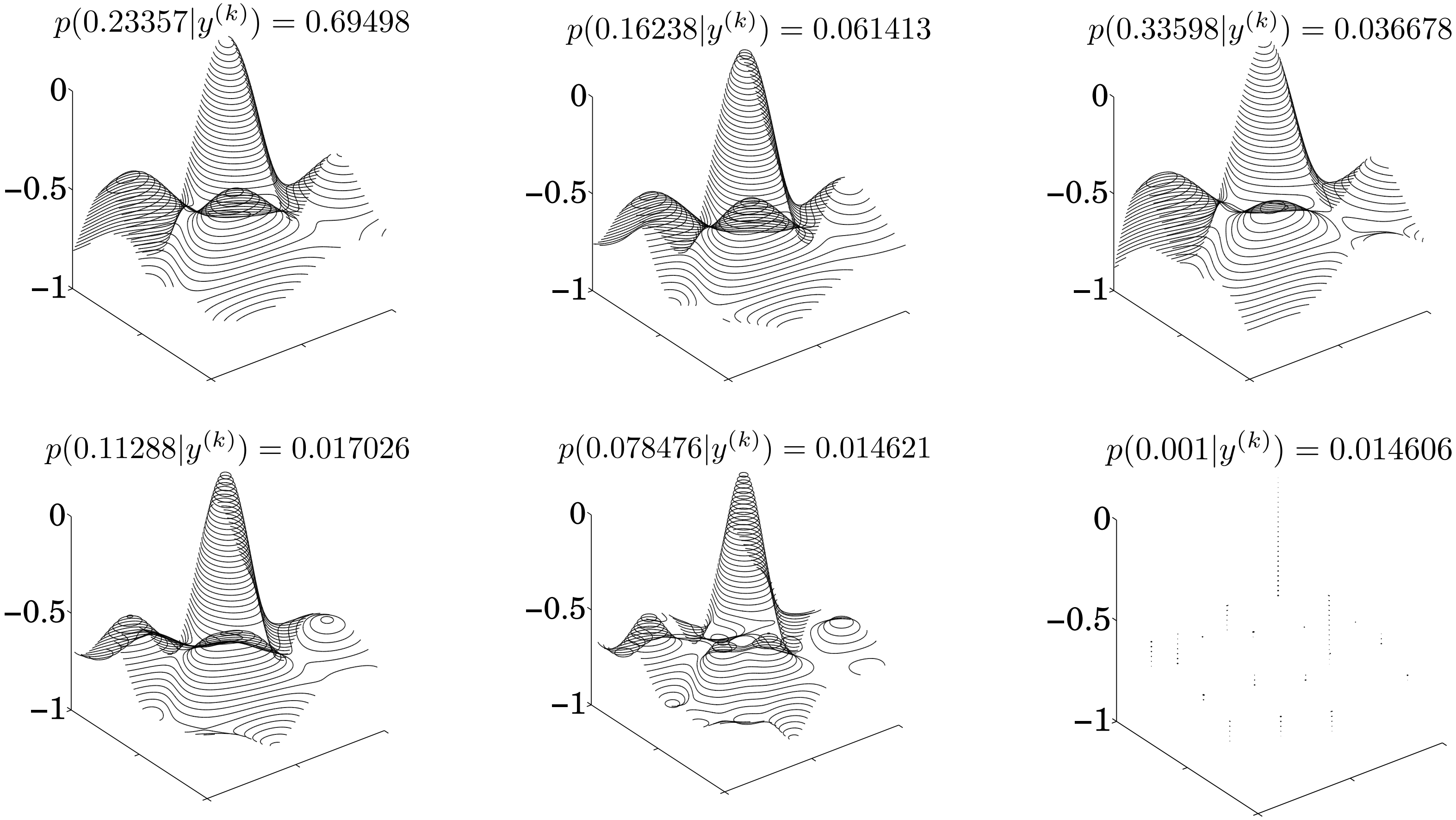


(a) Prior and posterior

(b) $\hat{Y}(x_1, x_2)$

(c) $s^2(x_1, x_2)$

Figure 20: Fully Bayesian model of the test function based on final uniform p.m.f. $p(\theta)$ (left). The domain of the p.m.f. has been expanded again. The distribution is now clear, as a coarse version of the distribution in Figure 17 is revealed. The GP model (middle) now shows some of the subtle features of the true function.



$p(0.23357|y^{(k)}) = 0.69498$

$p(0.16238|y^{(k)}) = 0.061413$

$p(0.33598|y^{(k)}) = 0.036678$

$p(0.11288|y^{(k)}) = 0.017026$

$p(0.078476|y^{(k)}) = 0.014621$

$p(0.001|y^{(k)}) = 0.014606$

Figure 21: Some of the favored modes generated by the proposed method. The negative of each surface is plotted for better visualization.

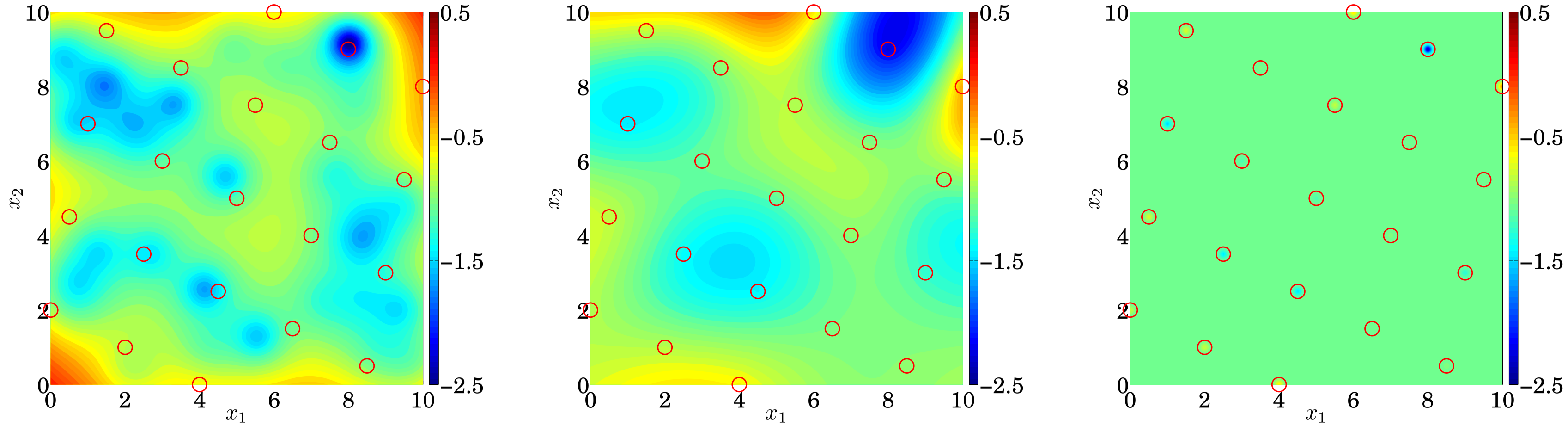


Figure 22: Comparison of proposed method with MLE. True function, left; fully Bayesian GP model generated by proposed method, middle; GP model generated by MLE, right.

Remark 2.2.1 Relation of fully Bayesian approach to MAP method

As $p(\theta)$ and hence $p(\theta|y^{(k)})$ become more populated, a p.m.f. such as that shown in Figure 17 arises. The maximum of the distribution is the most likely value, and this is precisely the value that is sought after in the maximum *a posteriori* (MAP) method, i.e.,

$$\hat{\theta}_{\text{MAP}} = \arg\max_{\theta} \left(p(\theta|y^{(k)}) \propto p(y^{(k)}|\theta)p(\theta) \right) \tag{2.35}$$

The MLE approach searches for the θ that maximize $p(y^{(k)}|\theta)$.

2.3 Radial Basis Functions

The material in this section is based on [44, 54, 57].

The radial basis function interpolation problem is to find a function of the form

$$\begin{aligned} \hat{y}(x) &= \sum_{i=1}^{k} \lambda_i \phi(\|x - x^{(i)}\|_2) + p(x) \\ &= \sum_{i=1}^{k} \lambda_i \phi(\|x - x^{(i)}\|_2) + \sum_{i=1}^{\mathfrak{p}} b_i p_i(x) \end{aligned} \tag{2.36}$$

that interpolates $\mathcal{S}$ at the observation locations $x^{(i)}$, $i = 1, \ldots, k$. The coefficients λ_i, $i = 1, \ldots, k$, are real numbers and $p_i(\cdot)$ is from $\mathcal{P}_n^d$, the space of polynomials of degree less than or equal to d in $\mathbb{R}^n$ with $\mathfrak{p} = \text{dim}\mathcal{P}_n^d$. The following choices of $\phi(\cdot)$ are considered:

$$\begin{aligned} &\phi(r) = r && \text{linear} \\ &\phi(r) = r^3 && \text{cubic} \\ &\phi(r) = r^2 \log r && \text{thin plate spline} \\ &\phi(r) = \sqrt{r^2 + \gamma^2} && \text{multiquadric} \\ &\phi(r) = e^{-\gamma r^2} && \text{Gaussian} \end{aligned} \tag{2.37}$$

where $r \geq 0$ and γ is a prescribed positive constant. The polynomial $p(\cdot)$ is required so that the linear system of equations that must be solved to obtain the interpolator is nonsingular. The coefficients of $\hat{y}(\cdot)$ in equation (2.36) are defined uniquely by the system

$$\begin{aligned} \hat{y}(x^{(i)}) &= y(x^{(i)}), \quad i = 1, \ldots, k \\ \sum_{i=1}^{k} \lambda_i p_j(x^{(i)}) &= 0, \qquad\quad j = 1, \ldots, d \end{aligned} \tag{2.38}$$

In matrix form, this is

$$\begin{bmatrix} \Phi & P \\ P^T & 0 \end{bmatrix} \begin{bmatrix} \lambda \\ b \end{bmatrix} = \begin{bmatrix} y^{(k)} \\ 0 \end{bmatrix} \tag{2.39}$$

where $\Phi_{ij} = \phi(\|x^{(i)} - x^{(j)}\|_2)$ for $i, j = 1, \ldots, k$, $P_{ij} = p_j(x^{(i)})$ for $i = 1, \ldots, k$ and $j = 1, \ldots, \mathfrak{p}$, $\lambda = [\lambda_1, \ldots, \lambda_k]^T$, and $b = [b_1, \ldots, b_{\mathfrak{p}}]^T$.

2.4 Kernel Machines

The material in this section is based on [94].

GP models are an example of a class of methods known as kernel machines. GP models are distinguished by the probabilistic viewpoint taken (see the references in [57, 94]). Support vector machines (SVMs), relevance vector machines (RVMs), splines, and least-squares classifiers are all examples of kernel machines [94].

The problem of inferring an underlying function $y(\cdot)$ from a finite and possibly noise corrupted dataset without any additional assumptions is ill-posed; any function that passes through the observations is acceptable. Under a Bayesian approach, a prior is assigned to a set of underlying functions, and given the observations, the posterior is derived. The main argument against this viewpoint is the non-rigorous process of assigning the prior. The *regularization* viewpoint addresses this assumption by investigating the smoothness of $y(\cdot)$. Consider the functional

$$J\left(y(x)\right) = \frac{\lambda}{2}\|y\|^2_{\mathcal{H}} + Q(T, y^{(k)}) \tag{2.40}$$

where T is a vector of target values to be predicted and λ is a scaling term that trades off the two terms. The first term is called the regularizer and represents the smoothness assumptions on $y(\cdot)$, and the second term is a data-fit term assessing the quality of the prediction $\hat{y}(x^{(i)})$ at the observations. The regularization method returns $\hat{y}(\cdot) = \arg\min_y J\left(y(x)\right)$, which can be viewed as the MAP estimate under the posterior. While the regularization solution gives a part of the GP solution, it suffers from the following limitations:

1. It does not characterize the uncertainty in the predictions, nor does it handle well any multimodality in the posterior.

2. The analysis is focused at approximating the first level of Bayesian inference, concerning predictions for $y(\cdot)$. It is not usually extended to the next level, e.g., to the computation of the marginal likelihood, which is useful for setting the parameters of the covariance function and for model comparison.

GP spline models are closely related to the regularization method. While they do not suffer from the same drawbacks as the regularization method, GP spline models result in a piecewise cubic polynomial mean function but with nonsmooth posterior samples.

SVMs were originally introduced for classification problems, then extended to deal with regression. The key concept is that of the ϵ-insensitive error function, defined as

$$g_\epsilon(z) = \begin{cases} |z| - \epsilon, & |z| \geq \epsilon \\ 0, & |z| < \epsilon \end{cases} \tag{2.41}$$

which is a data-fit assessment term that is used in manner similar to that of equation (2.40). A disadvantage of SVM regression is that it does not have a clear probabilistic interpretation. In addition, the use of equation (2.41) as the data-assessment term discards observations close to the mean function $\hat{y}$. This effect is motivated by the desire to obtain a sparse solution and hence reduce the fitting time, but for expensive functions, discarding observations may not be desirable.

RVMs are a special case of GPs, but with degenerate covariance functions that depend on the observations:

$$\Bbbk(x^{(i)}, x^{(j)}) = \sum_{m=1}^{k} \alpha_m^{-1} \phi_m(x^{(i)}) \phi_m(x^{(j)}) \tag{2.42}$$

where the basis functions $\phi_m(x^{(i)})$ are centered on the k observations, i.e., $\phi_m(x^{(i)}) = \phi_m(\|x^{(i)} - x^{(m)}\|)$. This dependence violates the Bayesian interpretation. Fitting the RVM is similar to fitting some GP models: maximize the marginal likelihood with respect to the hyperparameters. The optimization process may lead to a significant number of the α_m tending towards infinity, effectively pruning the corresponding basis functions from the covariance function (2.42). The pruned basis functions are considered irrelevant in the model and the surviving basis functions are termed relevance vectors. Typically the number of relevance vectors is less than the number of training cases, thus it will be faster to fit a RVM than a GP. There is one serious drawback to RVMs: points that lie far away from the relevance vectors will be predicted with variance close to zero. Thus, RVMs provide desirable computational properties, but at the expense of modeling integrity.

CHAPTER III

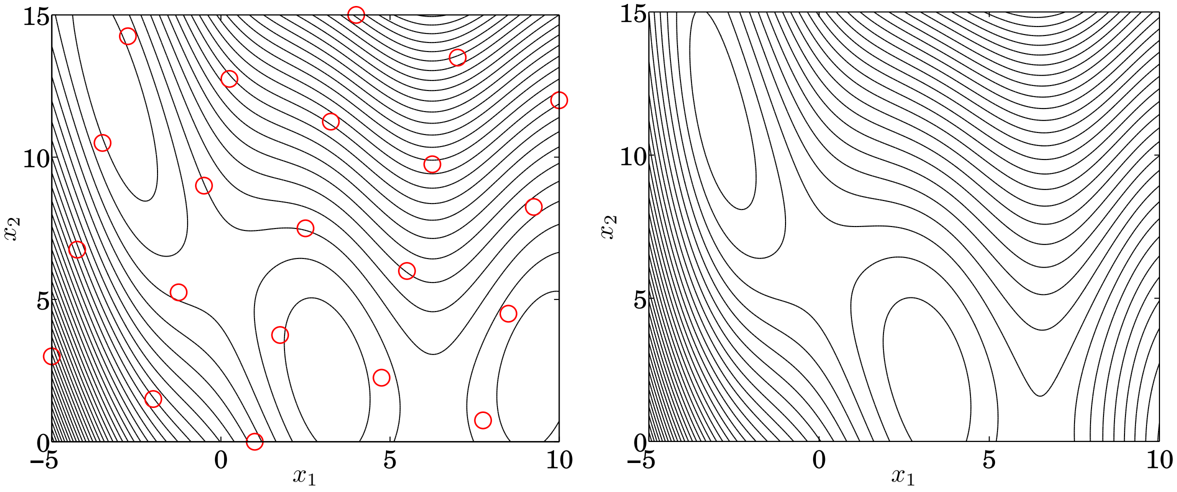
INFILL SAMPLING CRITERIA

3.1 Overview Of Existing Algorithms

Within the engineering community, surrogate models have gained popularity as a way to develop computationally inexpensive representations of expensive functions. This is evident from the number of recent journal papers and textbooks published on the subject [40, 57, 54, 56, 108, 44, 97, 79]. The promise of surrogate model-based global optimization is depicted in Figure 23, where a GP model has been used to reconstruct the Branin function; the reconstruction is nearly indistinguishable from the true function.

The simplest surrogate model-based approach for global optimization is to sample the design space, fit a global surrogate model, and sample new points by selecting those that minimize the model. The model is then updated and the process is repeated until some stopping criterion is met. However, this approach fails to search globally and is likely to miss the global minimum or can converge to a point that may not even be a critical point of the original function [54]. Thus, ISCs for surrogate model-based global optimization must balance exploration with exploitation.

At the highest level, surrogate models can be differentiated based on whether they are interpolating or non-interpolating. Some surrogate models are capable of both regression and interpolation with similar computational effort. For selecting iterates, a key distinction is between one-stage and two-stage methods. Most current approaches are two-stage methods. In the first stage, a surrogate model is fit to the available data. In the second stage, an ISC based on the surrogate model is maximized in order to guide the selection of the next iterate. The main problem with two-stage methods



(a) Branin function with Latin hypercube samples

(b) Gaussian process model

Figure 23: Illustration of surrogate model-based approach.

is that the initial sample may give a misleading picture of the function; as a result, one may underestimate the error in the surrogate model and either terminate the search prematurely or place too much emphasis on local search. In one-stage approaches, no surrogate model is fit through the data. Instead, surrogate modeling mechanics are used to evaluate hypotheses about the existence of points with certain values. Table 4 presents a taxonomy of six surrogate model-based global optimization methods from the literature, which will be reviewed in this chapter. Four of the methods are Bayesian (P-algorithm, EGO, osEGO, FB-EGO), while the remaining two algorithms typically use RBFs (RBF-G and CORS). A seventh algorithm (not listed), is known as `DIRECT`. This is not a surrogate model-based method but rather a Lipschitzian algorithm which has been shown to be competitive.

surrogate model-based global optimization has four advantages [54]. First, this technique often requires the fewest function evaluations of all competing methods. This is possible because a search algorithm can move to areas of the model that predict minima, rather than having to take steps along some trajectory like local methods. Second, some surrogate model-based approaches can provide credible stopping rules based on confidence intervals. Third, the surrogate model-based approach provides an inexpensive approximation to the expensive function. Finally, through proper experimental design, statistical analyses can be performed on the model to determine the most important variables.

Table 4: Overview of existing algorithms (adapted from [54]).

Surrogate Model	**Search Method**	
	***One-stage methods*:** evaluate hypotheses about minimum using surrogate model	***Two-stage methods*:** fit a model then find the next iterate by solving an auxiliary problem
Interpolating	■ RBF-G [44] ■ osEGO [91]	■ EGO [56] ■ CORS [97] ■ P-Algorithm [122] ■ FB-EGO [10]
Non-interpolating	■ osEGO [91]	■ EGO [56] ■ CORS [97] ■ P-Algorithm [122] ■ FB-EGO [10]

3.2 Improvement-Based Infill Sampling Criteria

3.2.1 Probability Of Improvement

The most popular surrogate model-based global optimization algorithms are heuristically motivated by an improvement function which is a random variable that describes the probability of obtaining a new function value $y(x)$ that is better than some target value $y^T \leq y_{\min}$, where $y_{\min} \triangleq \min_{1\leq i\leq k} y(x^{(i)})$ is the current best function value over k function evaluations made so far. Let $Y \sim \mathcal{N}(\hat{Y}, s^2)$ be the random variable that models the uncertainty about a function's value at x (the dependence of $\hat{Y}$ and s^2 on x is implied). The improvement below y^T at a point x is $I = (y^T - Y)_+$, where

$(z)_+ = \max(z, 0)$. $I(\cdot)$ is a random variable because $Y(\cdot)$ is a random variable. An ISC for GP-based algorithms can be constructed by taking moments of the improvement function and selecting the next iterate $x^{(k+1)}$ as the maximizer of the auxiliary function. Taking the first moment of the improvement function results in the popular ISC known as the probability of improvement, i.e.,

$$\begin{aligned} \mathbb{E}[I^0(x)] &= \int_{-\infty}^{y^T} p(Y)dY \\ &= \mathrm{P}(Y(x) \leq y^T) \end{aligned} \tag{3.1}$$

Algorithms which employ the ISC (3.1) are collectively known as P-algorithms. The P-algorithm seeks the next iterate using a one-step lookahead method on the conditional probability[1] to fall below a given level y^T:

$$x^{(k+1)} = \arg\max_{x \in A} \mathrm{P}\left(Y(x) < y^T | \hat{Y}(x^{(i)}) = y(x^{(i)}), i = 1, \ldots, k\right) \tag{3.2}$$

Thus, the P-algorithm maximizes the probability of improvement below a target y^T at each iterate (see Figure 25). This result has been developed both heuristically [62] and axiomatically [114, 123]. Žilinskas [122] develops the P-algorithm for a univariate objective function that is modeled by a Wiener process. The selection of y^T at each step is critical to the success of the algorithm, where $y^T \ll y_{\min}$ leads to a fairly global search and y^T closer (but still less than) $y_{\min}$ leads to a fairly local search. Kushner [62] chooses

$$y^T = y_{\min} - \epsilon \tag{3.3}$$

with monotonically decreasing ϵ. Žilinskas [122] chooses

$$y^T = y_{\min} - \frac{1}{2}\left(\max_{1 \leq i \leq k} y(x^{(i)}) - y_{\min}\right) \tag{3.4}$$

[1] The conditioning event $\hat{y}(x^{(i)}) = y(x^{(i)}), i = 1, \ldots, k$ represents the noise-free information gained in the observations $1, \ldots, k$ and is sometimes omitted in the equations in the literature. In such cases it is understood that the next observation depends on all previous information.

but it is shown in a later paper [21] that the strategy of Kushner leads to better convergence rates. Specifically, Calvin and Žilinskas [21] choose

$$\epsilon_i = i^{-1+\delta} \tag{3.5}$$

for some small positive $\delta < 1$ at each iteration i.

The majority of work on the P-algorithm [62, 122, 123, 124, 22] has been for one-dimensional multimodal optimization and extending the axiomatic P-algorithm to the n-dimensional case has proved difficult [22, 125, 114, 113, 81, 82, 69, 124]. In this research, the heuristic motivation for the P-algorithm is extended to the multidimensional case using GPs [54]. The search criterion is defined by (3.2). Jones [54] choses a set of targets T and solves (3.2) multiple times per iteration. The results are then clustered and a few points are chosen to proceed. However, this approach proved to be costly and did not produce better results than choosing a single target y^T based on a strategy in [44]. In both cases, values of y^T cycle such that the search proceeds from global to local. The performance of the P-algorithm on the one-dimensional test function

$$y(x) = x\left(\sin(10x+1) + 0.1\sin 15x\right) \tag{3.6}$$

is shown in Figure 24.

Remark 3.2.1 Numerical underflow in GP-based P-algorithm [41]

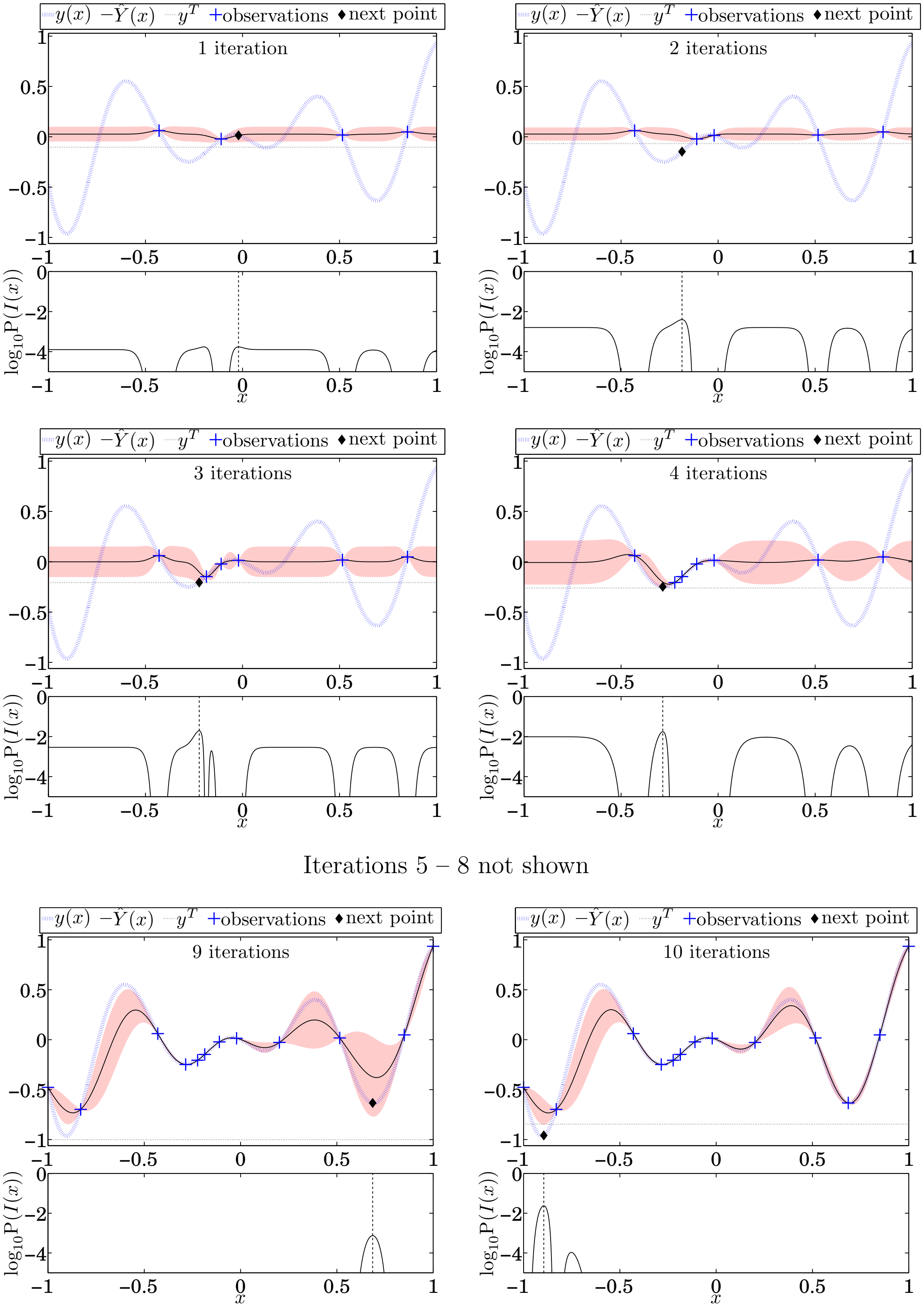
For a GP, the probability of improvement in problem (3.2) is calculated as

$$\begin{aligned} \mathrm{P}(\hat{y}(x) < y^T) &= \Phi(u) && (3.7)\\ &= \frac{1}{2}\left(1 + \operatorname{erf}(a)\right) && (3.8) \end{aligned}$$

where $u \triangleq (y^T - \hat{Y})/s$ and $a = u/\sqrt{2}$. Typically, the target values y^T will be large and negative and so $\operatorname{erf}(\cdot) \to -1$. This often leads to numerical underflow and $\mathrm{P}(\hat{y}(x) < y^T) = 0$. When $a \ll -1$, $\operatorname{erf}(a)$ can be expressed using an asymptotic expansion [3, eq. 7.2.14] and equation (3.8) becomes

$$\mathrm{P}(Y(x) < y^T) = \left(\frac{1}{2\sqrt{\pi}} \sum_{n=0}^{\infty} \frac{(-1)^n (2n-1)!!}{2^n} a^{-(2n+1)} \right) \exp(-a^2) \tag{3.9}$$

Problem (3.2) can be then solved over the logarithmic space without underflow problems. □



Iterations 5 – 8 not shown

Figure 24: Selected iterations of a GP-based P-algorithm on test function (B.11). Shaded area denotes an uncertainty region of ±2s

3.2.2 Expected Improvement

Perhaps a more powerful way to make use of the improvement function is the maximize its expectation over A and select the next iterate as the maximizer of the expected improvement. An improvement in the best function value $y_{\min}$ found so far will be obtained if $Y < y_{\min}$, so $I = (y_{\min} - Y)_+$ in this case. The expected improvement (EI) is then

$$\mathbb{E}_I(x) \triangleq \mathbb{E}\left[(y_{\min} - Y)_+\right] \tag{3.10}$$

where $\mathbb{E}_I(x)$ denotes the expectation of I as a function of x. Let $u \triangleq (y_{\min} - \hat{Y})/s$ and let $v \triangleq (Y - \hat{Y})/s$. Then $I = (s(u-v))_+$ and

$$\mathbb{E}_I(x) = \int_{-\infty}^{u} s(u-v)p(v)dv \tag{3.11}$$

For a GP, the expectation of the improvement can be written in closed form as

$$\mathbb{E}_I(x) = \begin{cases} s[u\Phi(u) + \phi(u)], & s > 0 \\ 0, & s = 0 \end{cases} \tag{3.12}$$

where $\phi(\cdot)$ and $\Phi(\cdot)$ are the standard normal density and distribution functions, respectively. The first term in equation (3.12) is the predicted difference between the current minimum and the prediction $\hat{Y}(x)$, penalized by the probability of improvement below $y_{\min}$. Hence, it is large when $\hat{Y}(x)$ is likely to be smaller than $y_{\min}$. The second term is large when the error $s(x)$ is large, i.e., when there is a high degree of uncertainty about whether $y(x)$ will be better than $y_{\min}$. Thus, the first term controls the local search while the second term controls the global search. The balance between exploitation and exploration is automatically adjusted by the parameters in equation (3.12) and no target values are required (see Figure 25). Recent work [118, 20] has characterized some of the convergence properties of expected improvement algorithms.

Algorithms which employ the ISC (3.10) are collectively known as EGO-type algorithms, after the Efficient Global Optimization (EGO) algorithm [56] which popularized the EI criterion. EGO-type algorithms seek the next iterate using a one-step lookahead method by maximizing EI:

$$x^{(k+1)} = \arg\max_{x \in A} \mathbb{E}_I(x) \tag{3.13}$$

Under certain conditions, (3.13) converges to the global minimum of any continuous function [69]. The performance of the EGO algorithm on the one-dimensional test function (B.11) is shown in Figure 26. Initially, EGO wastes function evaluations identifying a local minimum instead of exploring the design space. This is a consequence of the low uncertainty estimate provided by the model, which has been by MLE. By the 6th iteration (not shown), EGO has approximated the local minimum well and then explores areas outside the local minimum. In the context of computationally expensive programs, this behavior is unsatisfactory.

Remark 3.2.2 Numerical underflow in EGO

Numerical underflow can occur in the EI criterion when $s(\cdot)$ becomes small. The arguments in Remark 3.2.1 can be extended to EGO to obtain

$$\mathbb{E}_I(x) = s\left(\frac{u}{2\sqrt{\pi}}\sum_{n=0}^{\infty}\frac{(-1)^n(2n-1)!!}{2^n}a^{-(2n+1)} + \frac{1}{\sqrt{2\pi}}\right)\exp(-a^2) \tag{3.14}$$

Problem (3.13) can be then solved over the logarithmic space without underflow problems. □

The EI criterion eliminates the need to specify a target value y^T as in the probability of improvement. However, this also eliminates control over exploration and exploitation. If the initial interpolating model is inaccurate compared to the actual function, EGO would be slow to converge because the user cannot control how much emphasis to place on the local versus global search in equation (3.12) [110]. Sobester et al. [110] propose a weighted expected improvement function $\mathbb{E}_I(x; w)$ where a

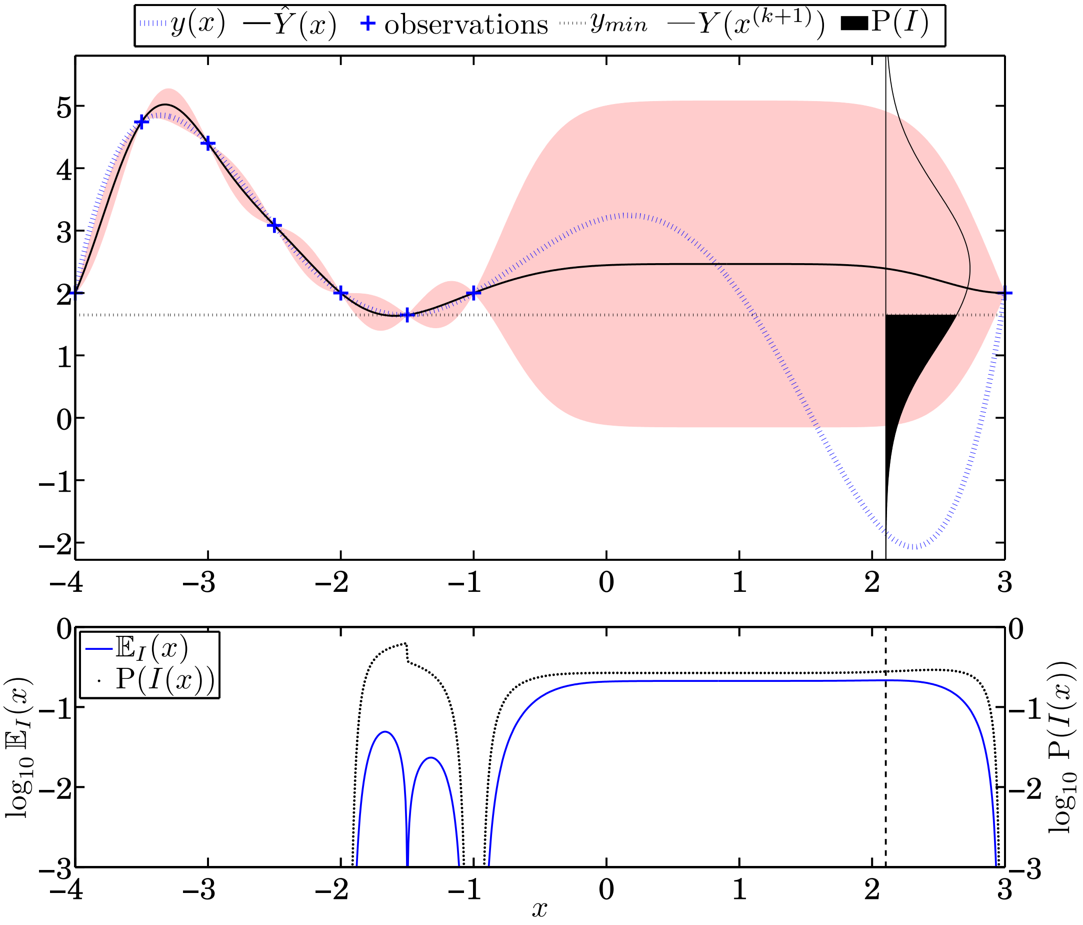


Figure 25: Graphical interpretation of $I(x) = (y_{\min} - Y)_+$ at the location $\arg\max_{x \in A} \mathbb{E}_I(x)$ along with true function and GP regression, top figure; $\mathbb{E}_I(x)$ and $P(I(x))$, bottom figure. Shaded area denotes an uncertainty region of $\pm 2s$.

weighting factor $w \in [0, 1]$ shifts the balance between local and global search:

$$\mathbb{E}_I(x; w) = \begin{cases} w(y_{\min} - \hat{Y})\Phi(u) + (1 - w)s\phi(u), & s > 0 \\ 0, & s = 0 \end{cases} \tag{3.15}$$

A correction is also required in equation (3.15); when $\mathbb{E}_I(x; w) < 0$, the value of $\mathbb{E}_I(x; w)$ must be set to 0.

If knowledge of the complexity of the function being minimized is known *a priori*, w can be selected accordingly. In the case of a black-box function where no information is known, Sobester et al. [110] recommend cycling through the available range of global-local balances, e.g., $w = \langle 0.1, 0.3, 0.5, 0.7, 0.9\rangle$.

Schonlau et al. [108] generalize the EI criterion to allow more control over how global the search will be. The generalized EI includes an integer-valued parameter g. The larger the value of g, the more globally the algorithm will tend to search. The generalized improvement is given by

$$I(x; g) = \begin{cases} s^g(u - v)^g, & u > v, s > 0 \\ 0, & \text{else} \end{cases} \tag{3.16}$$

For $g = 1$, the improvement criterion used by Jones et al. [56], $I = (y_{\min} - Y)_+$, is recovered. The generalized expected improvement can be written as

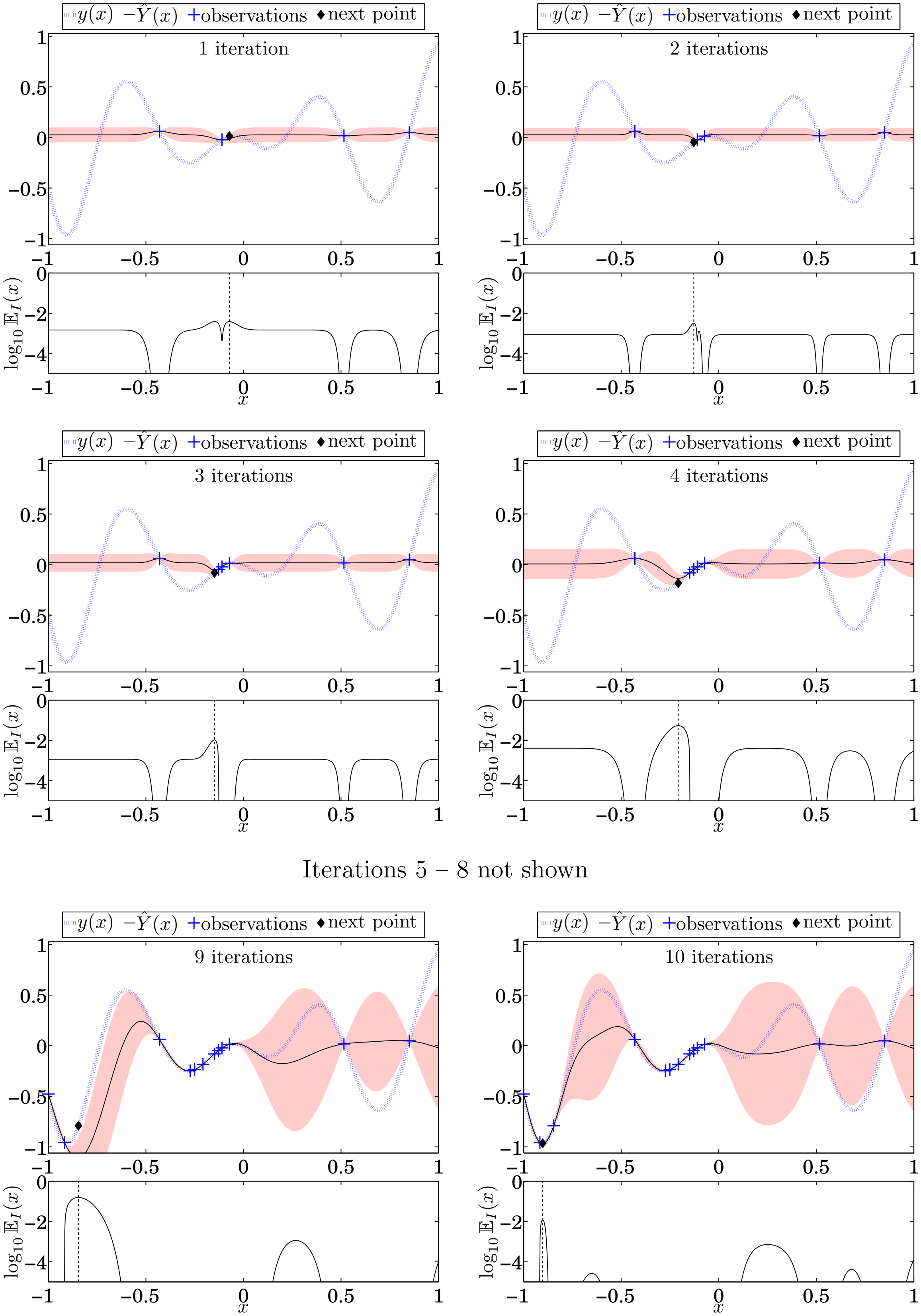
$$\mathbb{E}_I(x; g) = s^g \sum_{k=0}^{g} (-1)^k \binom{g}{k} u^{g-k} T_k \tag{3.17}$$

where T_k for $k > 1$ can be computed recursively from

$$T_k = -\phi(u)u^{k-1} + (k - 1)T_{k-2} \tag{3.18}$$

with initial values

$$T_0 = \Phi(u) \qquad \text{and} \qquad T_1 = -\phi(u) \tag{3.19}$$



Iterations 5 – 8 not shown

Figure 26: Selected iterations of the EGO algorithm on test function (B.11).

3.2.3 Fully Bayesian Expected Improvement

It was demonstrated in §3.2.2 that in the case of deceptive functions, the MLE-based EGO can waste many function evaluations searching locally due to the underestimated uncertainty in the GP model. This is because EGO is a two-stage approach, i.e., the hyperparameters of the GP model are estimated by maximum likelihood and then plugged into the EI sampling criterion, which is then maximized to obtain the location of the next observation. Benassi et al. [10] advocate a fully Bayesian (FB) approach to address this problem.

The development of the fully Bayesian EI criterion follows from §2.2.5. The conditional EI criterion for this problem can be derived from equation (3.11) with $u \triangleq (y_{\min} - \mu)/\gamma_k$:

$$\mathbb{E}_{I|\theta}(x) = \gamma_k(x) \left[u T_{\eta_k}(u) + \frac{\eta_k + u^2}{\eta_k - 1} t_{\eta_k}(u) \right] \tag{3.20}$$

where $t_\eta(\cdot)$ and $T_\eta(\cdot)$ are the standard t density and distribution functions, respectively, with η degrees of freedom. Note that the conjugacy of the inverse Gamma and normal distributions has made an analytic derivation possible. In general, equation (2.20) or the more general (2.14) must be approximated [78].

$\mathbb{E}_{I|\theta}(x)$ is a random variable until a value for θ has been selected. Following the Bayesian methodology, the uncertainty in θ is marginalized to obtain:

$$\begin{aligned}\mathbb{E}_I(x) &= \mathbb{E}_\theta \left[\mathbb{E}_{I|\theta}(x) \right] \\ &= \int_\theta \mathbb{E}_{I|\theta}(x) p(\theta | y^{(k)}) d\theta \end{aligned} \tag{3.21}$$

Note that the plug-in EI criterion of §3.2.2 can be seen as an approximation of the fully Bayesian criterion (3.21), i.e.,

$$\int_\theta \mathbb{E}_{I|\theta}(x) p(\theta | y^{(k)}) d\theta \approx \mathbb{E}_{I|\hat{\theta}}(x) \tag{3.22}$$

which is justified if $p(\theta|y^{(k)})$ is concentrated enough around the maximum likelihood estimates $\hat{\theta}_{\text{MLE}}$.

Remark 3.2.3 Numerical underflow in FB-EGO

The following asymptotic expansion for the t-distribution has been derived in [112]:

$$T_{\eta_k}(a) = -(1 + a^2/\eta_k)t_{\eta_k}(a)/a \sum_{n=0}^{\infty} \frac{(-1)^n(2n-1)!!\eta_k^n}{(\eta_k+2)(\eta_k+4)\cdots(\eta_k+2n)a^{2n}} \tag{3.23}$$

By substituting $a = -u$ in equation (3.20) and using (3.23), the following expression for the fully Bayesian EI criterion conditional on θ can be written:

$$\begin{aligned} \mathbb{E}_{I|\theta}(x) =& \gamma_k(x) \left[-a(1 - T_{\eta_k}(a)) + \frac{\eta_k + a^2}{\eta_k - 1} t_{\eta_k}(a) \right] \\ =& \gamma_k(x) \left[-(1 + a^2/\eta_k) \sum_{n=0}^{\infty} \frac{(-1)^n(2n-1)!!\eta_k^n}{(\eta_k+2)(\eta_k+4)\cdots(\eta_k+2n)a^{2n}} \right. \\ & \left. + \frac{\eta_k + a^2}{\eta_k - 1} \right] t_{\eta_k}(a) \end{aligned} \tag{3.24}$$

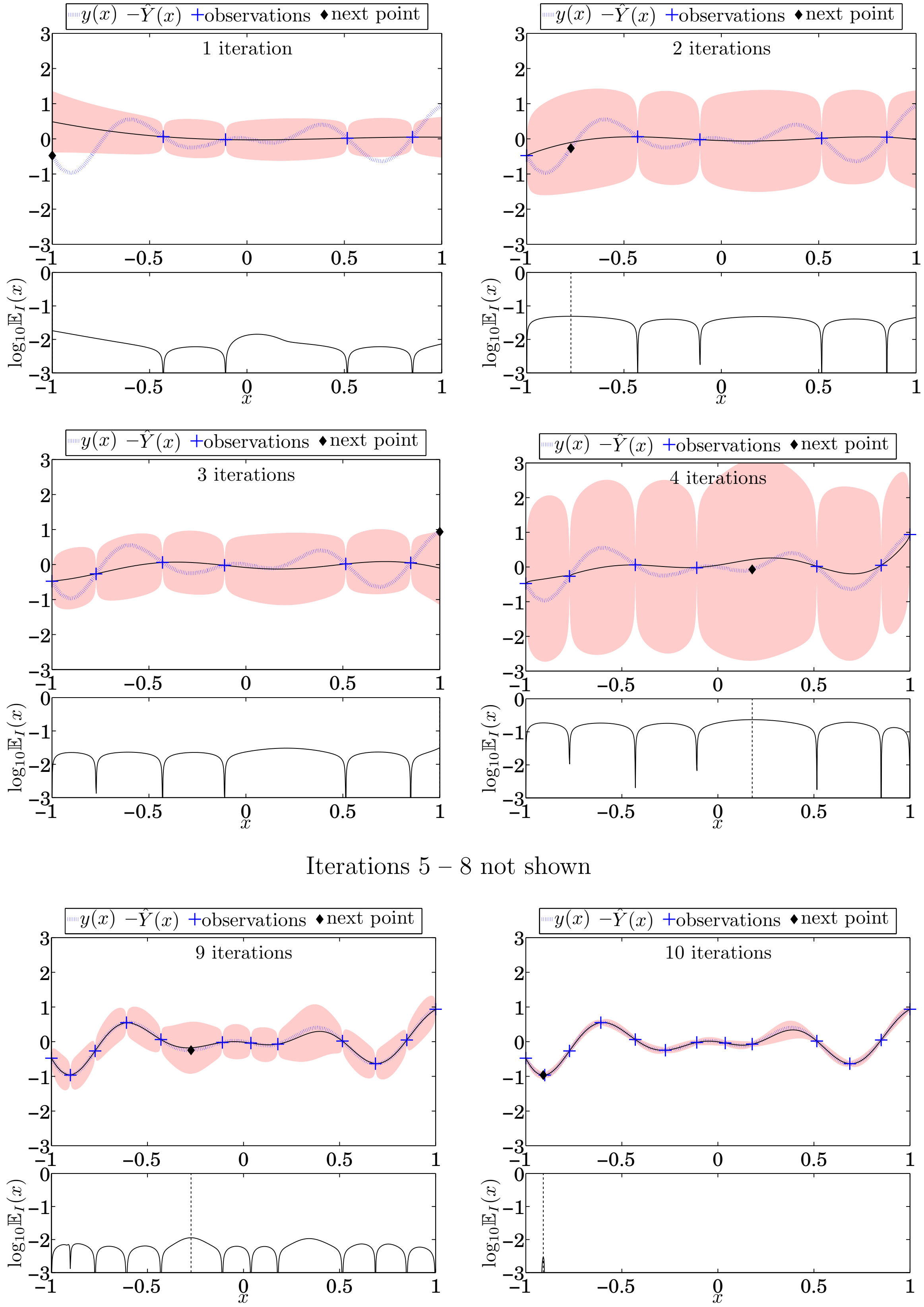
Unfortunately, this expression can only be used when one mode θ is considered, i.e., equation (3.22). In the fully Bayesian case, equation (3.21) must be used and there does not appear any way to take advantage of the asymptotic distribution to search over the log-space. □

Figure 27 shows the results of fully Bayesian EGO on test function (B.11). In contrast with the original EGO algorithm, the uncertainty in the hyperparameter θ and the variance σ^2 has been marginalized into the posterior predictive distribution, resulting in wider confidence bands. This forces the algorithm to search globally at first instead of locally around $y_{\min}$.

3.3 One-Stage Methods

3.3.1 One-Stage EGO

Thus far, all the EGO-based algorithms reviewed are two-stage methods, i.e., a surrogate model is fit to the sample set and the next sample point is found by maximizing an ISC. It may seem counterintuitive to solve one global optimization problem by solving another, but the optimization of the ISC is far easier than that of the original



Iterations 5 – 8 not shown

Figure 27: Selected iterations of the fully Bayesian EGO algorithm on test function (B.11). Shaded area denotes an uncertainty region of $\pm 2s$.

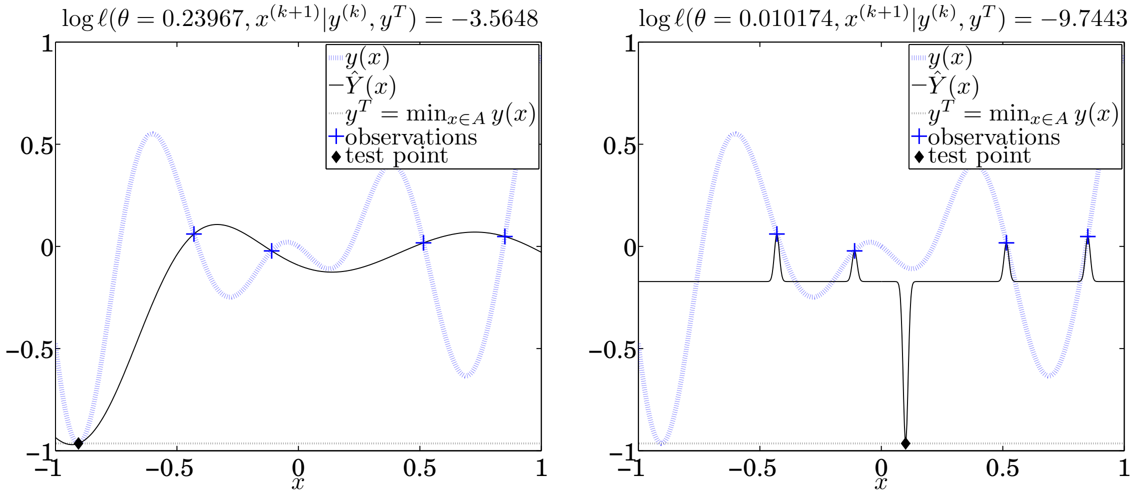


Figure 28: Illustration of one-stage methods. Likelihood of y^T occurring at the global minimizer, left; likelihood of y^T occurring at some other test point, left.

problem. This is the appeal of two-stage approaches. A drawback is that the ISC depends on the estimates of the hyperparameters, which may lead to poor decisions. In contrast, a one-stage method includes the calculation of $x^{(k+1)}$ in the MLE calculation (2.15), bypassing the dependence on previous hyperparameter estimates. This is accomplished by assuming that the GP interpolates the point $x^{(k+1)}$ with unknown function value $y(x^{(k+1)})$, thereby conditioning the likelihood of the sample set to this hypothesis. One-stage methods can lead to more accurate computations of $x^{(k+1)}$, but require the solution of a more difficult auxiliary problem, as well as some heuristic strategy that assigns values to $y(x^{(k+1)})$. Figure 28 illustrates the concept of one-stage methods. A target value $y^T = \min_{x \in A} y(x)$, i.e., the minimum of the true function, is selected. In general this is not possible since $y(\cdot)$ and specifically the global minimum of $y(\cdot)$ is unknown. Two candidate locations $x^{(k+1)}$ are selected and the logarithm of the conditional likelihood (discussed below) is used as the credibility criterion. The hypothesis that y^T occurs at $x^{(k+1)} = x^*$, i.e., the global minimizer of $y(x)$, in Figure 28 is more credible than the hypothesis that it occurs at some other test point shown.

Quttineh and Holmström [91] propose the one-stage EGO algorithm osEGO. The likelihood function of the observed data conditional upon the hypothesis that $\hat{y}$ interpolates $(x^{(k+1)}, y^T)$ is

$$\ell(\theta, x^{(k+1)}|y^{(k)}, y^T) = (2\pi)^{-n/2} \left|\mathbf{C}\right|^{-1/2} \exp\left[-\frac{1}{2}(\overline{y} - \overline{\mathbf{F}}\beta)^T \mathbf{C}^{-1}(\overline{y} - \overline{\mathbf{F}}\beta)\right] \tag{3.25}$$

where

$$\mathbf{C} = \mathbf{K} - \mathbf{k}_{k+1}\Bbbk(x^{(k+1)}, x^{(k+1)})^{-1}\mathbf{k}_{k+1}^T \tag{3.26}$$

$$\overline{y} = y^{(k)} - \mathbf{k}_{k+1}\Bbbk(x^{(k+1)}, x^{(k+1)})^{-1}y^T \tag{3.27}$$

$$\overline{\mathbf{F}} = \mathbf{F} - \mathbf{k}_{k+1}\Bbbk(x^{(k+1)}, x^{(k+1)})^{-1}\mathbf{f}_{k+1}^T \tag{3.28}$$

The vector $\mathbf{k}_{k+1}$ is $\mathbf{k}_0$ augmented with $\Bbbk(x, x^{(k+1)})$. Thus, for isotropic covariance functions with one hyperparameter θ, the problem of determining the $x^{(k+1)}$ becomes

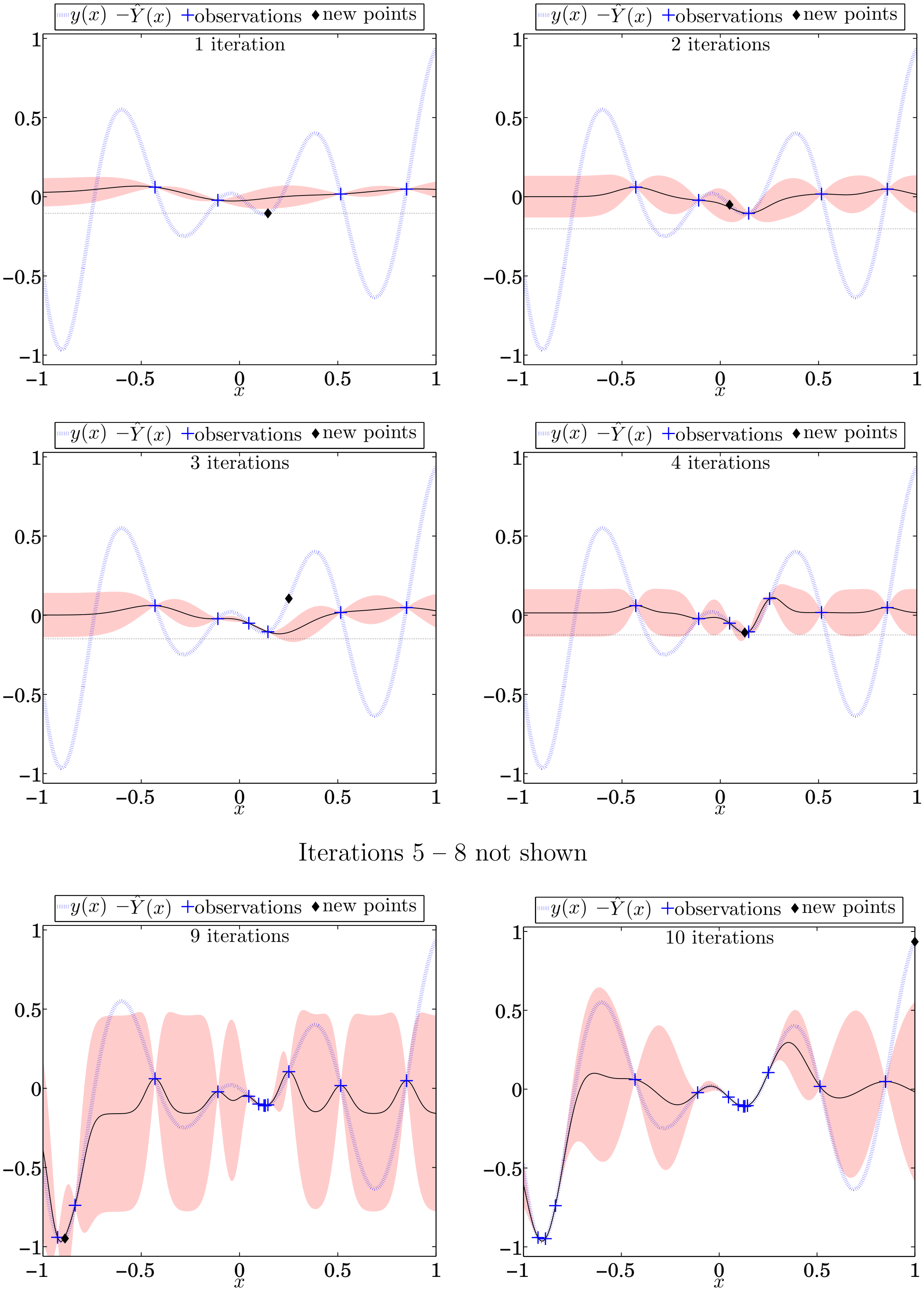
$$\begin{aligned} x^{(k+1)} = \ & \underset{\substack{\theta\in\mathbb{R} \\ x\in\mathbb{R}^n}}{\arg\max} && \log \ell(\theta, x|y^{(k)}, y^T) \\ & \text{subject to} && c_i(x) \geq 0, \qquad i = 1, \ldots, m \\ & && \theta > 0 \\ & && x^\ell \leq x \leq x^u \end{aligned} \tag{3.29}$$

Note that the inequality constraints $c_i(\cdot) \geq 0$ must now be included in the problem statement since the maximization is also over x. The strategy from [44] can once again be used to assign values to y^T.

Numerical results [91] indicate that osEGO is able to solve more problems to global optimality, but requires more function evaluations. This is the cost for the additional robustness provided by one-stage approaches. Its performance on the test function (B.11) is shown in Figure 29. The primary disadvantage of osEGO is that it is still a likelihood-based approach.

3.3.2 Method Of Gutmann

Gutmann [44] employs the following general surrogate modeling technique to develop a global optimization algorithm. Let $\mathcal{A}$ be the linear space of functions and assume



Iterations 5 – 8 not shown

Figure 29: Selected iterations of the osEGO algorithm on test function (B.11).

that for $s \in \mathcal{A}$, $\sigma(s)$ is a measure of the "bumpiness" of s. Assume that the sample set $\mathcal{S}$ has been evaluated on $\mathcal{D} = \{x^{(1)}, \ldots, x^{(k)}\}$. A target value y^T is chosen that can be regarded as an estimate of the minimum of $y(x)$ (see Algorithm 2 for Gutmann's method for selecting y^T). For each $x \notin \mathcal{D}$, let $\hat{y} \in \mathcal{A}$ be defined by the interpolation conditions

$$\begin{aligned} \hat{y}(x^{(i)}) &= y(x^{(i)}), \qquad i = 1, \ldots, k \\ \hat{y}(x) &= y^T \end{aligned} \tag{3.30}$$

The new sample point $x^{(k+1)}$ is chosen to be the value of x that minimizes $\sigma(\hat{y})$, $x \notin \mathcal{D}$. Gutmann 2001 argues that the "least bumpy" curve interpolating $\mathcal{S} \cup y^T$ will give the most reasonable minimum value at x. The algorithm is described next.

Consider a radial basis function $\phi(\cdot)$ from (2.37) and $d \geq d_0$. Let $p_i(\cdot), i = 1, \ldots, \mathfrak{p}$ be a basis of $\mathcal{P}_n^d$, the space of polynomials in $\mathbb{R}^n$ of degree less than or equal to d, where $\mathfrak{p} = \dim \mathcal{P}_n^d$. Assume $x^{(1)}, \ldots, x^{(k)}$ have been chosen to satisfy

$$q \in \mathcal{P}_n^d \quad \text{and} \quad q(x_i) = 0, i = 1, \ldots, k, \implies q \triangleq 0 \tag{3.31}$$

Let the function

$$s_k(x) = \sum_{i=1}^{k} \lambda_i \phi(\|x - x^{(i)}\|) + \sum_{i=1}^{\mathfrak{p}} b_i p_i(x) \tag{3.32}$$

interpolate $\mathcal{S}$ through $\mathcal{D}$. The goal is to determine $x^{(k+1)}$. For a target value y^T and a point $x \in A \setminus \mathcal{D}$, the RBF $\hat{y}(\cdot)$ that satisfies (3.30) can be written as

$$\hat{y}(x) = s_k(x) + \left(y(x^{(k)}) - s_k(x)\right) \ell_k(\xi, x) \tag{3.33}$$

where $\ell_k(\xi, x)$ is the radial basis function solution to the interpolation conditions

$$\begin{aligned} \ell_k(\xi, x^{(i)}) &= 0, \qquad i = 1, \ldots, k \\ \ell_k(\xi, \xi) &= 1 \end{aligned} \tag{3.34}$$

Therefore, $\ell_k(\xi, x)$ can be expressed as

$$\ell_k(\xi, x) = \sum_{i=1}^{k} \alpha_i(\xi) \phi(\|x - x^{(i)}\|) + \mu_k(\xi) \phi(\|x - x^{(i)}\|) + \sum_{i=1}^{\mathfrak{p}} b_i(\xi) p_i(x) \tag{3.35}$$

Let

$$u(\xi) = [\phi(\|\xi - x^{(1)}\|), \ldots, \phi(\|\xi - x^{(k)}\|)]^T$$

and

$$\pi(\xi) = [p_1(\xi), \ldots, p_{\mathfrak{p}}(\xi)]^T$$

Then, the coefficients of $\ell_k(\xi, x)$ are defined by the equations

$$\begin{bmatrix} \Phi & u(\xi) & P \\ u(\xi)^T & \phi(0) & \pi(\xi)^T \\ P^T & \pi(\xi) & 0_{\mathfrak{p}\times\mathfrak{p}} \end{bmatrix} \begin{bmatrix} \alpha(\xi) \\ \mu_n(\xi) \\ b(\xi) \end{bmatrix} = \begin{bmatrix} 0_k \\ 1 \\ 0_{\mathfrak{p}} \end{bmatrix} \tag{3.36}$$

where $\alpha(\xi) = [\alpha_1(\xi), \ldots, \alpha_k(\xi)]^T \in \mathbb{R}^k$, $b(\xi) = [b_1(\xi), \ldots, b_{\mathfrak{p}}(\xi)]^T \in \mathbb{R}^{\mathfrak{p}}, \mu_k(\xi) \in \mathbb{R}$, and 0_k and $0_{\mathfrak{p}}$ denote the zero column vectors in $\mathbb{R}^k$ and $\mathbb{R}^{\mathfrak{p}}$, respectively. Gutmann uses the square of the semi-norm $\langle \hat{y}, \hat{y} \rangle$ of the interpolant (3.35) as the measure of "bumpiness" and derives the formula

$$\langle \hat{y}, \hat{y} \rangle = \langle s_k, s_k \rangle + (-1)^{d_0+1} \mu_k(\xi) \left(y^T - s_k(\xi)\right)^2 \tag{3.37}$$

with integer $d_0 \geq 1$. Define the function $g_n : A \setminus \mathcal{D} \to \mathbb{R}$ as the difference

$$g_k(\xi; y^T) = \langle \hat{y}, \hat{y} \rangle - \langle s_k, s_k \rangle = (-1)^{d_0+1} \mu_k(\xi) \left(y^T - s_k(\xi)\right)^2 \tag{3.38}$$

which is nonnegative. Since $\langle \hat{y}, \hat{y} \rangle$ is independent of ξ, the required minimization of $\langle \hat{y}, \hat{y} \rangle$ and the minimization of $g_k(\xi; y^T)$ are equivalent. The choice of y^T determines the location of $x^{(k+1)}$. If $\min_{\xi \in A} s_k(\xi) \leq y^T \leq \max_{\xi \in A} s_k(\xi)$, then $g_k(\xi; y^T) = 0$ can be achieved and the algorithm will search locally near existing sample points. However, if $y^T \leq \min_{\xi \in A} s_k(\xi)$, then $x^{(k+1)}$ will search more globally. In particular, the choice $y^T = -\infty$ requires the minimization of the function $(-1)^{d_0+1} \mu_k(\xi)$, which leads to a global search in A. The RBF-based global optimization algorithm (RBF-G) by Gutmann is listed in Algorithm 2[2].

[2] There are several heuristic intricacies in the target selection system which have not been outlined here; see [44, 98] for details

Remark 3.3.1 The function g_k is infinitely differentiable on $A \setminus \mathcal{D}$ but it is not defined at the interpolation points. If $y^T = \min_{\xi \in A} s_k(\xi)$ and if $s_k(x^{(i)}) > y^T, i = 1, \ldots, k$, then the global minimizers of g_k are the global minimizers of s_k. Thus one can minimize s_k, which is defined on the whole of A, to obtain $x^{(k+1)}$. If $y^T < \min_{x \in A} s_k(x)$, however, then, $g_k(\xi; y^T) \to \infty$ as $\xi \to x^{(i)}, i = 1, \ldots, k$. Let $h_k : A \to \mathbb{R}$ be defined as

$$h_k(\xi; y^T) = \begin{cases} 1/g_k(\xi; y^T), & \xi \notin \mathcal{D} \\ 0, & \xi \in \mathcal{D} \end{cases} \tag{3.39}$$

The maximization of h_k on A is equivalent to the minimization of g_k. Further, h_k is infinitely differentiable on $A \setminus \mathcal{D}$. □

The following theorem from [44] describes the additional conditions necessary for convergence of Algorithm 2 by establishing the density of the sequence of generated points $x^{(k+1)}$.

Theorem 3.3.2 [44] *Let $\phi(r) = r$, $\phi(r) = r^2 \log r$, or $\phi(r) = r^3$. Further, choose the integer d such that $0 \leq d \leq n$ in the linear case, $0 \leq d \leq n+1$ in the thin-plate spline case, $1 \leq d \leq n+2$ in the cubic case. Let $\{x^{(k)}\}, k \in \mathbb{N}$ be the sequence generated by Algorithm (2), and s_k be the RBF that interpolates $\{(x^{(i)}, y(x^{(i)}))\}, i = 1, \ldots, k$. Assume that for infinitely many $k \in \mathbb{N}$, the choice of target value y_k^T satisfies*

$$\min_{\xi \in A} s_k(\xi) - y_k^T > \tau \Delta_n^{\rho/2} \|s_k\|_\infty \tag{3.42}$$

where $\Delta_k := \min_{1 \leq i \leq k-1} \|x^{(k)} - x^{(i)}\|_2$, $\tau > 0$, and $0 \leq \rho \leq 1$ in the linear case and $0 \leq \rho \leq 2$ in the thin-plate spline and cubic cases. Then the sequence $\{x^{(k)}\}, k \in \mathbb{N}$ is dense in A.

An important result is that Gutmann's method reduces to the Wiener process-based P-algorithm in the special case $\phi(r) = r$ and $n = 1$.

Figure 30 illustrates the performance of RBF-G on test function (B.11). A cubic RBF model is used with linear polynomial regressors, i.e., $\phi(r) = r^3$ and $d_0 = 1$.

Algorithm 2 Gutmann-RBF algorithm.

```
Pick $\phi(\cdot)$ from (2.37) and $d \geq d_0$. In particular, $d_0 = 1$ in the cubic and thin plate spline cases, $d_0 = 0$ in the linear and multiquadric cases, and $d_0 = -1$ in the Gaussian case.
Choose points $\mathcal{D} = \{x^{(1)}, \ldots, x^{(k)}\} \subset A$ that satisfy (3.31) and generate the initial sample set $\mathcal{S}^{(k)} = \{y(x^{(1)}), \ldots, y(x^{(k)})\}$.
Set $i \leftarrow 1$, $k_0 = k$, and select the cycle length $N$.
while not converged do
    Compute the radial basis function $s_i$ by solving system (2.39).
    Let $\alpha$ be a permutation of $\{1, \ldots, k\}$ such that $y(x^{(\alpha_{(1)})}) \leq \cdots \leq y(x^{(\alpha_{(k)})})$.
    Select target values $y^T$ according to:
        $y_i^T = \min_{x \in A} s_i(x) - w_k \left( y(x^{\alpha(i_k)}) - \min_{x \in A} s_i(x) \right)$    (3.40)
    where
        $w_k = \left( \frac{\mathrm{mod}(N - (k - k_0), N + 1)}{N} \right)^2$
        $i_k = \begin{cases} i, & \mathrm{mod}(k - k_0, N + 1) = 0 \\ i_{k-1} - \lfloor (k - k_0)/N \rfloor, & \text{else} \end{cases}$
    Compute $x^{(k+1)}$ by solving
        $x^{(k+1)} = \arg \max_{\xi \in A \setminus \mathcal{D}} h_i(\xi; y^T)$    (3.41)
$\mathcal{D}^{(k+1)} \leftarrow \mathcal{D}^{(k)} \cup x^{(k+1)}$, $\mathcal{S}^{(k+1)} \leftarrow \mathcal{S}^{(k)} \cup y(x^{(k+1)})$, $i \leftarrow i + 1$, $k \leftarrow k + 1$
end while
```

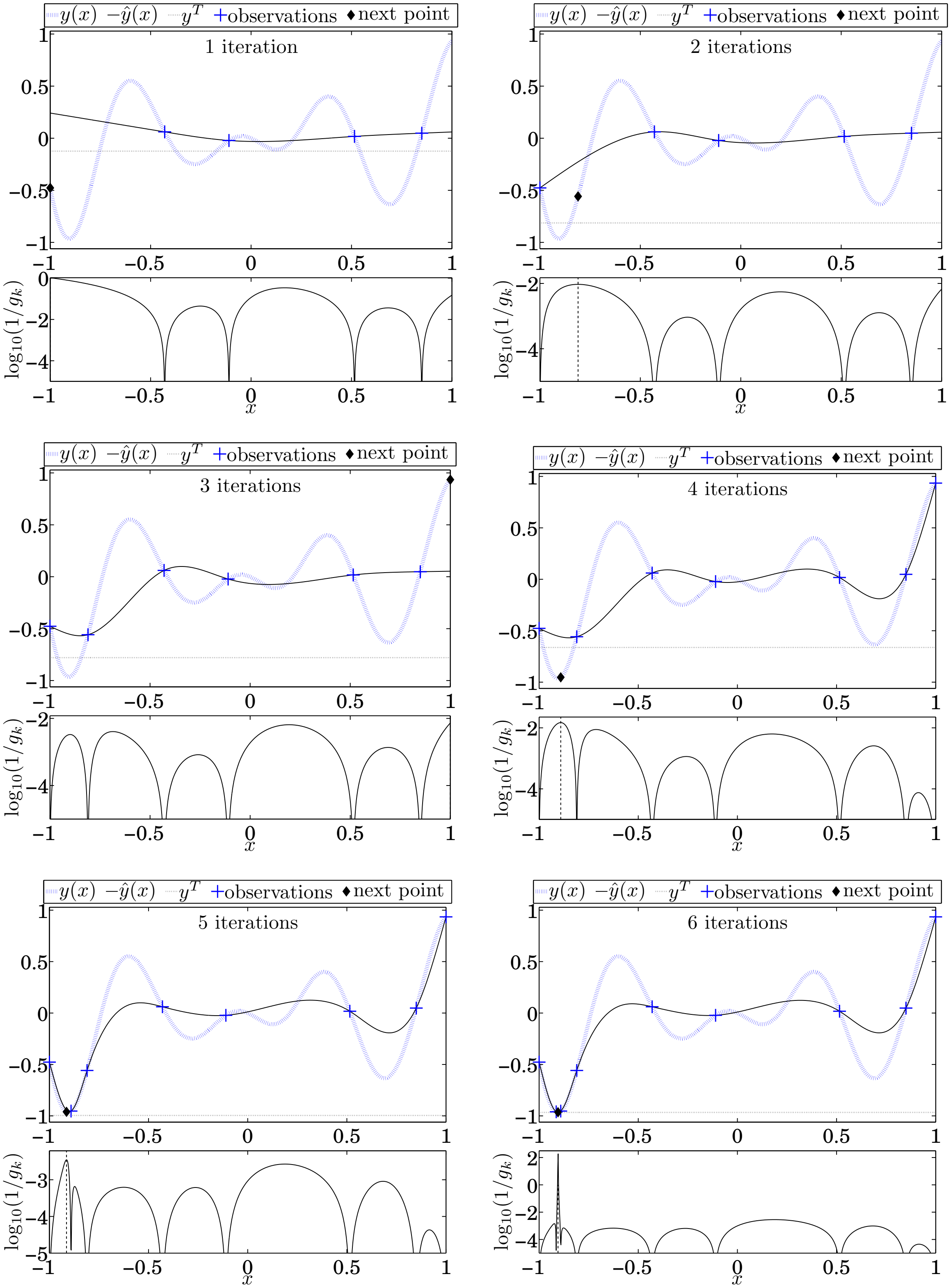


Figure 30: Selected iterations of the RBF-G algorithm on test function (B.11) with $\phi(r) = r^3$ and $d_0 = 1$.

3.4 Other Methods

3.4.1 CORS

Regis and Shoemaker [97] present the Constrained Optimization using Response Surfaces (CORS) method which searches for sample points by minimizing the current surrogate model $\hat{y}(x)$ subject to the bound constraints $x \in A$ and to the additional constraints that the next iterate should be some minimum distance from all previously sampled iterates. The point which is as far away as possible from any previously evaluated point is referred to as the maximin point Δ_i, and all sample points in CORS are required to be at least some fraction β_i of Δ_i from previously sampled points. For iteration i, the auxiliary problem is written as:

$$\begin{aligned} \underset{x \in A \subset \mathbb{R}^n}{\text{minimize}} \quad & \hat{y}(x) \\ \text{subject to} \quad & \|x - x^{(j)}\|_2 \geq \beta_i \Delta_i, \quad j = 1, \ldots, k \\ & x \in A \end{aligned} \tag{3.43}$$

where

$$\Delta_i = \max_{x \in A} \min_{1 \leq j \leq k} \|x - x^{(j)}\|_2 \tag{3.44}$$

Values of β_i are cycled starting from $\beta_i \approx 1$ and ending at $\beta_i \approx 0$, i.e., starting from a global search and ending in a local search. Regis and Shoemaker solve for the maximin point (3.44) by using a dense space-filling design and selecting the point that is farthest from any previously evaluated points. The auxiliary problem (3.43) is solved using a standard gradient-based optimizer. Regis and Shoemaker implement CORS with a RBF model as the surrogate model (termed CORS-RBF) and test two different cycles for β,

$$\text{SP1} = \langle 0.95, 0.25, 0.05, 0.03, 0 \rangle \quad \text{and} \quad \text{SP2} = \langle 0.9, 0.75, 0.25, 0.05, 0.03, 0 \rangle$$

Results [97] indicate that cycle SP1 is superior to SP2.

Figure 31 illustrates the performance of the CORS-RBF algorithm on test function (B.11). A cubic RBF model is used with linear polynomial regressors, i.e., $\phi(r) = r^3$

and $d_0 = 1$. The attractiveness of CORS is in its simplicity and computational speed: the surrogate model itself is the ISC and by minimizing the current surrogate model subject to constraints that balance exploration with exploitation, competitive performance with more sophisticated algorithms can be achieved. CORS is globally convergent and independent of the surrogate model chosen. However, CORS is sensitive to the cycle chosen for β and to the surrogate model that is chosen, which may require extensive testing to obtain satisfactory results.

3.4.2 `DIRECT`

All algorithms discussed thus far are surrogate model-based algorithms that compute one update point per iteration by maximizing an ISC which is based on the surrogate model or surrogate modeling mechanics. The *Di*viding *Rect*angles (`DIRECT`) algorithm of Jones et al. [53] takes a unique approach to satisfying the competing goals of the ISC. `DIRECT` is not a surrogate model-based algorithm; it is a method that, in each iteration in the solution of problem (NLP), computes one or more iterates using all possible weights on local versus global search (how this is done will be made clear shortly).

The bounds on the variables limit the search to an n-dimensional hypercube. `DIRECT` begins by trisecting this cube into smaller rectangles, each of which has a sampled point at its center. Figure 32 shows the first three iterations of `DIRECT` on a hypothetical two-variable problem. At the start of each iteration, the space is partitioned into rectangles. `DIRECT` then selects one or more of these rectangles for further search using a selection technique described in the next paragraph. Finally, each rectangle is trisected along one of its long sides, after which the center points of the outer thirds are sampled.

The key step in the algorithm is the selection of rectangles, since this determines how search effort is allocated across the space. As motivation, consider the extremes

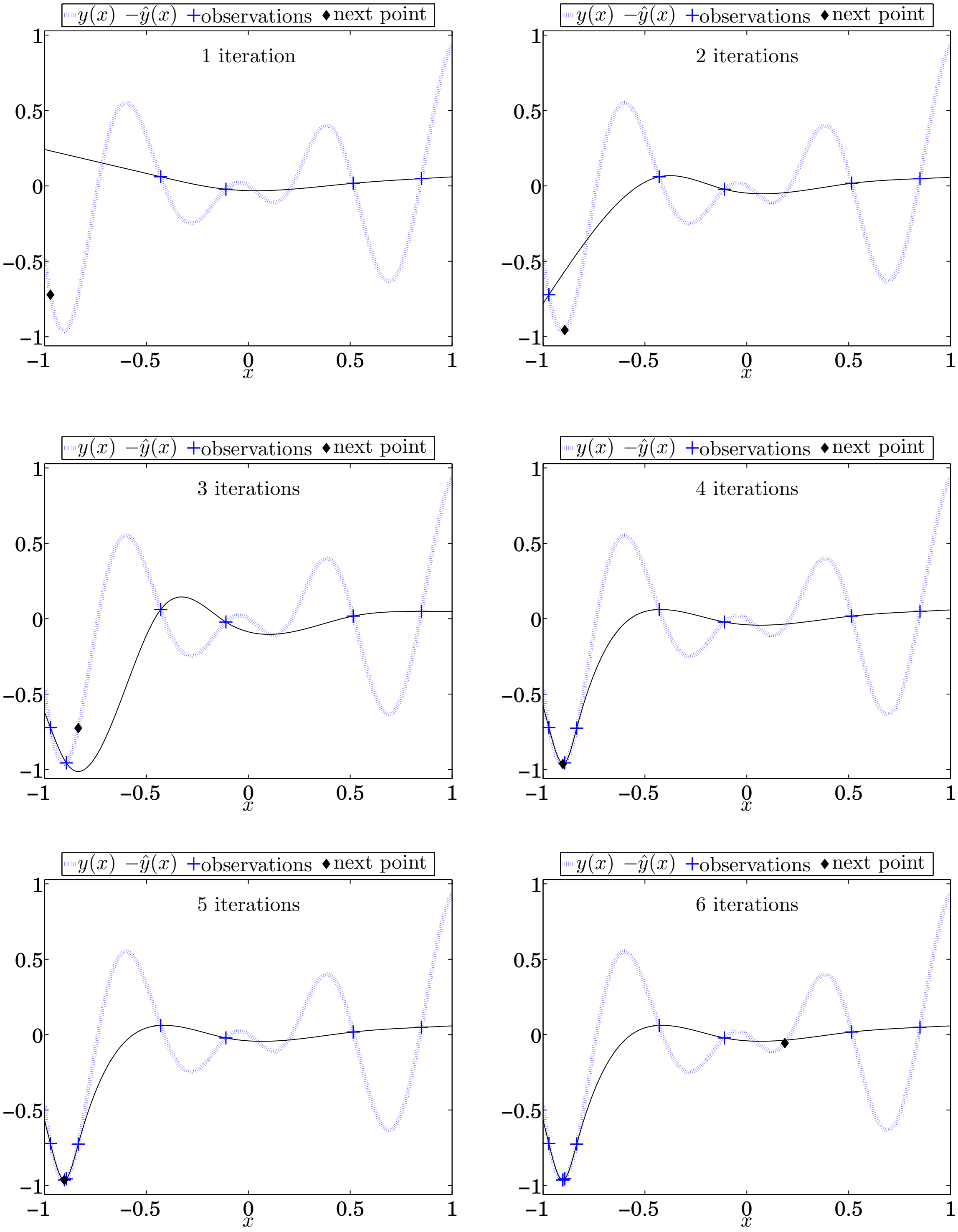


Figure 31: Selected iterations of the CORS-RBF algorithm on test function (B.11) with $\phi(r) = r^3$ and $d_0 = 1$.

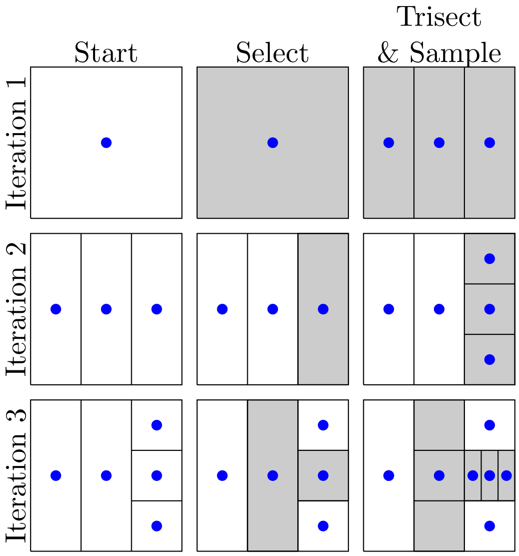


Figure 32: First three iterations of `DIRECT` on a hypothetical two-variable bound constrained problem.

of pure global search and pure local search. A pure global search would select one of the largest rectangles in each iteration in order to explore all parts of the design space uniformly. A pure local search would sample the rectangle with lowest function value in an attempt to improve the best solution, but may overlook a larger rectangle which could contain the global minimum. Selecting a single "best" rectangle would require a tuning parameter to control the global/local balance, but the algorithm would be extremely sensitive to this parameter. Due to the competing effects of the global and local search, it is possible to have multiple "best" solutions, which form a Pareto frontier between the rectangle's centerpoint value and the size of the rectangle, measured by its center-vertex distance. These are the rectangles that `DIRECT` selects to sample, and the strategy is illustrated in Figure 33. The performance of `DIRECT` on the Branin function (problem B-2) after $N = 20$ simplex gradients is shown in Figure 34.

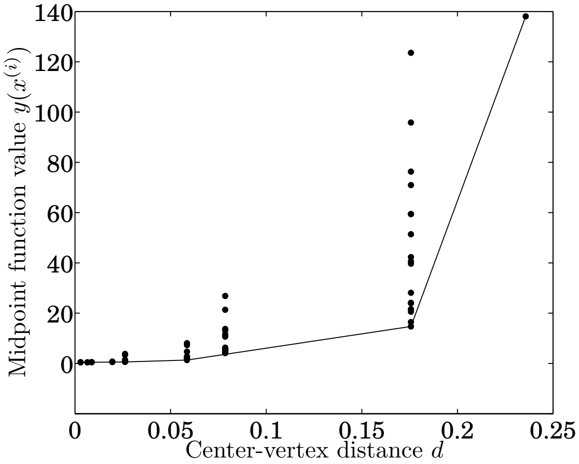


Figure 33: Pareto-based rectangle selection scheme for Branin function (problem B-2) after $N = 20$ simplex gradients.

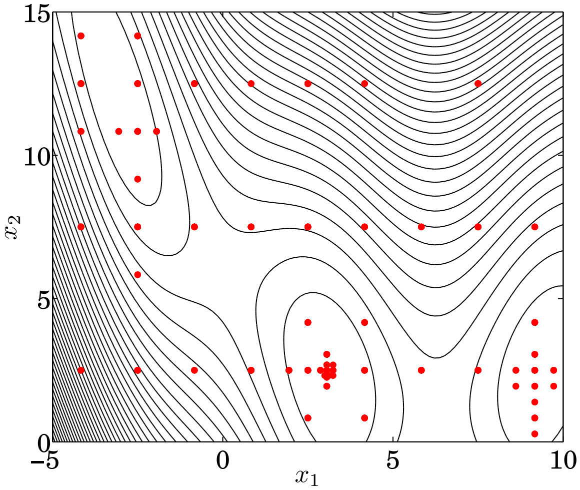


Figure 34: Performance of `DIRECT` on Branin function (problem B-2) after $N = 20$ simplex gradients.

CHAPTER IV

METHODS FOR HANDLING NONLINEAR CONSTRAINTS

This chapter discusses methods for handling nonlinear constraints. Penalty methods are discussed first, which are popular methods for transforming constrained problems into unconstrained problems. Penalty methods can be used with any of the surrogate models reviewed in Chapter 2, although there are certain limitations in the case of GP models. For GP models, specialized constraint handling techniques have been developed, which are discussed after the penalty methods. Some additional specific strategies follow.

4.1 Penalty Methods

Penalty methods attempt to solve the general nonlinear program (NLP) by combining the constraints and the objective function into a scalar penalty function that weighs each constraint (or its violation) by a penalty parameter. Some penalty functions obtain a solution to (NLP) as a sequence of unconstrained optimizations that converges to the solution of (NLP) while others solve (NLP) exactly as a single unconstrained problem. Some popular methods of each type are reviewed in the literature [75, 117].

A challenge associated with the use of penalty functions is the update of the penalty parameter(s) μ. Some penalty functions are *inexact*, i.e., their minimizer is generally not the same as the solution of (NLP) for any positive value of μ, while others are *exact*, which means that for certain choices of the penalty parameter(s), a single minimization with respect to x can yield the solution of (NLP); see [75, Theorem 17.3]. Bjorkman and Holmström [15] implement the ℓ_1 nonsmooth exact penalty function of the general form

$$\phi(x;\mu) = y(x) + \sum_{i\in\mathcal{E}} \mu_i\,|c_i(x)| + \sum_{i\in\mathcal{I}} \mu_i(-c_i(x))_+ \tag{4.1}$$

to solve an expensive global optimization problem. The general framework is outlined in Algorithm 3 and Figure 35 shows the transformation of a nonlinearly constrained problem into an unconstrained problem via (4.1). This approach is adopted to allow some of the unconstrained algorithms to handle constraints. In particular, osEGO, RBF-G, and CORS use this formulation.

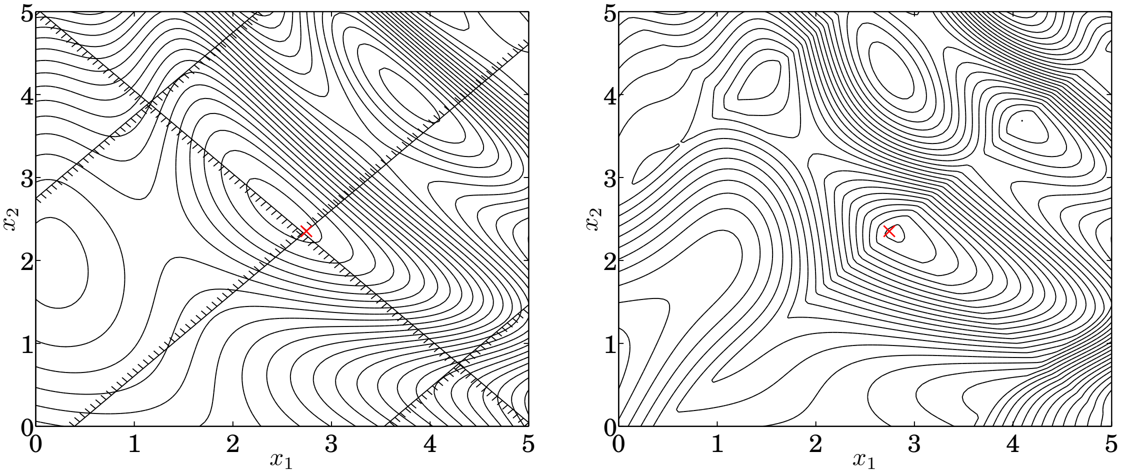


Figure 35: Example of ℓ_1 nonsmooth penalty function. Original constrained problem, left; unconstrained problem as a result of applying (4.1), right, with $\mu_1 = 11.53$ and $\mu_2 = 17.81$.

It remains to define a strategy for updating the penalty parameter(s) μ. Bjorkman and Holmström [15] choose μ to scale the objectives and constraints to be of unit order. However, it appears that the rate of change of the constraints is the more critical issue [117]. Each constraint must be scaled so that its *gradient* is of the same order of magnitude as the gradient of the objective function. This serves two purposes. First, it ensures that the curvature of $\phi(\cdot;\mu)$ is not dominated by a single constraint, which conditions the problem better for gradient-based optimization. Second, it makes $\phi(\cdot;\mu)$ less sensitive to μ. Informal testing indicates that setting μ by this strategy

Algorithm 3 General framework for surrogate model-based global optimization of nonlinearly constrained problems using penalty function.

```
Select initial design $\mathcal{D}^{(1)} = \{x^{(1)}, \ldots, x^{(k)}\} \subset A$ and compute the initial sample sets
$\mathcal{S}^{(1)} = \{y(x^{(1)}), \ldots, y(x^{(k)})\}$ and $\mathcal{C}_j^{(1)} = \{c_j(x^{(1)}), \ldots, c_j(x^{(k)})\}$ for $j = 1, \ldots, m$
Set $i \leftarrow 1$, $\mu_j = 0$, $j = 1, \ldots, m$
while not converged do
    Evaluate the penalty function (equation (4.1)) on $\mathcal{D}^{(i)}$ to obtain $\Phi^{(i)}$
    Fit model to $\mathcal{D}^{(i)}$ and $\Phi^{(i)}$
    Compute the next sample point by solving
        $x^{(k+1)} = \arg \max_{x \in A} ISC(x; \mu)$
    Update: $\mathcal{D}^{(i+1)} \leftarrow \mathcal{D}^{(i)} \cup x^{(k+1)}$, $\mathcal{S}^{(i+1)} \leftarrow \mathcal{S}^{(i)} \cup y(x^{(k+1)})$, $\mathcal{C}_j^{(i+1)} \leftarrow \mathcal{C}_j^{(i)} \cup c_j(x^{(k+1)})$
    for $j = 1, \ldots, m$, $\Phi^{(i+1)} \leftarrow \Phi^{(i)} \cup \phi(x^{(k+1)}; \mu)$, $i \leftarrow i+1$, $k \leftarrow k+1$
    Optionally update $\mu_j$, $j = 1, \ldots, m$
end while
```

is sufficient for (4.1) to be exact for most problems, while larger values of μ are detrimental to the conditioning of the surrogate model. Implementation details are discussed in §7.4.

The primary disadvantage of using classical penalty methods in Bayesian global optimization is that these methods do not exploit the uncertainty in the Bayesian model of (NLP), i.e., they are not Bayesian.

4.2 Bayesian Methods

4.2.1 Method Of Schonlau et al.

Schonlau et al. [108] propose a modification to the generalized improvement function, equation (3.16), to take the inequality constraints $c_i(x) \geq 0$ into account. Let $C_i(x) \sim \mathcal{N}(\hat{C}_i(x), s_i^2(x))$ denote the GP model for constraint i, i.e., assume that separate GP models for the objective and constraints have been constructed using the sample set.

Then the generalized improvement subject to the constraints can be defined as

$$I_c(x;g) = \begin{cases} (y_{\min} - Y)^g\,, & Y < y_{\min}, C_i \geq 0, i = 1, \ldots, m \\ 0, & \text{otherwise} \end{cases} \tag{4.2}$$

Here, $y_{\min}$ is defined as the minimum *feasible* value of the objective among all the points in the sample set. If the sample set contains no feasible points, the sample with the least constraint violation can be used and then this value can be switched to the minimum feasible value as soon as one becomes available. The expected value of $I_c(x;g)$ can be written as a multidimensional integral of the form

$$\mathbb{E}_{I_c}(x;g) = \int_0^{\infty} \cdots \int_0^{\infty} \int_{-\infty}^{y_{\min}} (y_{\min} - Y)^g p(Y, C_1, C_2, \ldots, C_m) dY dC_1 dC_2 \cdots dC_m \tag{4.3}$$

where $p(Y, C_1, C_2, \ldots, C_m)$ denotes the joint probability distribution function of $Y, C_1, C_2, \ldots, C_m$. By assuming that the random variables are statistically independent (and hence uncorrelated), equation (4.3) simplifies to

$$\mathbb{E}_{I_c}(x;g) = \mathbb{E}_I(x;g)\mathrm{P}(C_1 \geq 0)\mathrm{P}(C_2 \geq 0) \cdots \mathrm{P}(C_m \geq 0) \tag{4.4}$$

which is just the generalized expected improvement for the unconstrained case multiplied by the probability of feasibility of each constraint. The probabilities in equation (4.4) are computed from

$$\begin{aligned} \mathrm{P}(C_i \geq 0) &= 1 - \Phi(-u_i) \\ &= \Phi(u_i), \qquad i = 1, \ldots, m \end{aligned} \tag{4.5}$$

where $u_i = \hat{C}_i(x)/s_i(x)$. Similarly, the method can be used to extend the P-algorithm to constrained problems by replacing $\mathbb{E}_I(x;g)$ in equation (4.4) with $\mathrm{P}(Y(x) \leq y^T)$, resulting in the constrained probability of improvement. Figure 36 shows the performance of the constrained EI criterion on a multimodal problem with two nonlinear inequality constraints. The global minimum is located to within 1% accuracy by the third function evaluation, in addition to the twenty-one maximin LHD samples used to construct the models.

Since separate GP models must be constructed for the objective and constraints, poor performance can result when these are fit by MLE. Each model is subject to the inadequacies of MLE discussed in §sec:inadequacies and poor models can result for any or all of the functions under this method. When this occurs, nonsensical sample placement can occur due to the multiplicative nature of the constrained EI criterion (4.4). The fully Bayesian approach addresses these inadequacies and it is inferred that the performance of a nonlinearly constrained algorithm employing fully Bayesian models for the objective and constraints will exhibit superior performance than likelihood-based approaches.

Remark 4.2.1 Numerical underflow in constrained EI

Equation (4.4) is numerically ill-suited for computer implementation due to the product of probabilities that may approach 0 for a large portion of A. The results of Remarks 3.2 and 3.2.2 can be combined to give an expression for the logarithm of equation (4.4), which can then be maximized without underflow problems. □

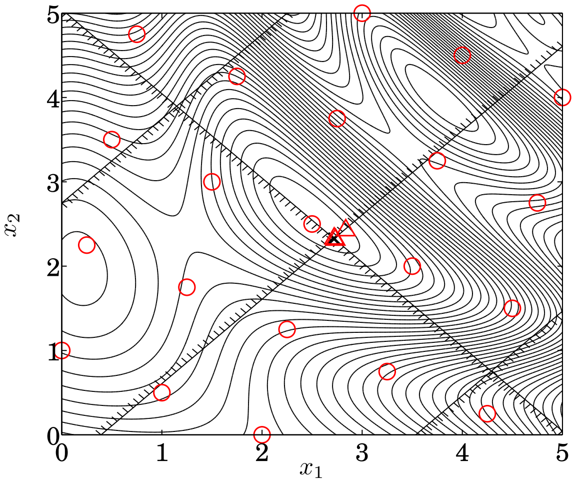


Figure 36: Performance of the constrained EGO algorithm, equation (4.4). True function and constraint with initial sample sites (○) and first three iterates (△).

4.2.2 Expected Violation

Audet et al. [8] extend the EI concept to constraints by introducing the *expected violation* (EV). Define the violation of the constraint GP as

$$V_i(x) = (-C_i(x))_+ , \qquad i = 1, \ldots, m \tag{4.6}$$

The expected violation is then

$$\mathbb{E}_{V_i}(x) = \mathbb{E}[(-C_i(x))_+], \qquad i = 1, \ldots, m \tag{4.7}$$

EI and EV are combined to form the Constrained, Balanced, Local-Global Search (CBLGS) algorithm. A dense LHD is used to sample the design space and all points where the EV falls below a user-specified threshold are accepted for calculation of EI. The expensive function is then evaluated at the points in this set with the highest EI values. The main drawback with the EV criterion is that it must be used in conjunction with an improvement-based criterion, otherwise a pure EV-based algorithm will search for points that are furthest from the constraint boundaries, i.e., points that minimize the expected violation.

4.3 DIRECT

The unconstrained version of `DIRECT` was introduced in §3.4.2. The method has been extended to handle nonlinear inequality constraints [55]; in this research, equality constraints are handled via the ℓ_1 penalty function (equation (4.1)). While the unconstrained `DIRECT` algorithm has been thoroughly tested [53, 14, 28, 35, 42], no results have been found in the open literature that assess the performance of the constrained `DIRECT` algorithm. This method will be included in the competing algorithms for nonlinearly constrained problems.

The key to handling inequality constraints in `DIRECT` is to define an auxiliary function that combines the objective and constraint functions. Let $c_j(x^{(r)})$ denote

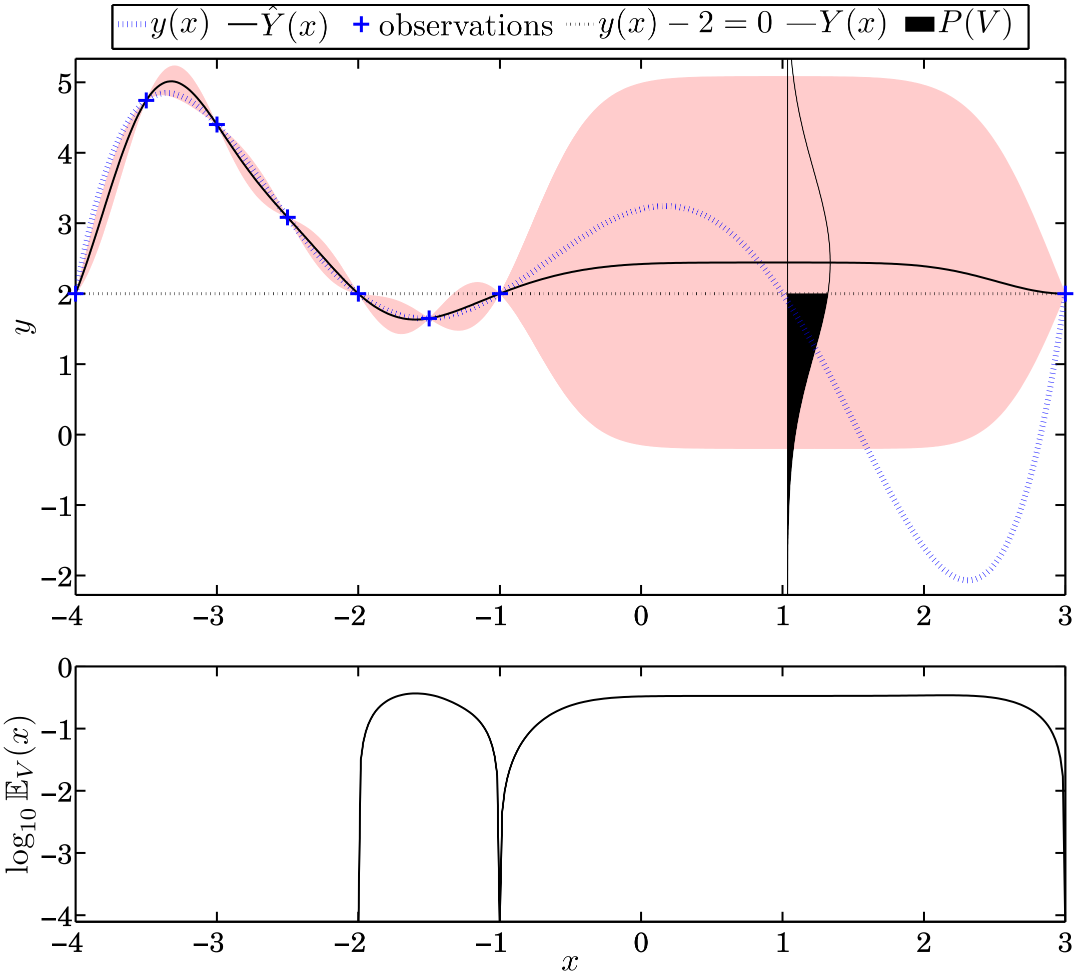


Figure 37: Graphical interpretation of $V(x)$ at the location $\arg\max_{x\in A}\mathbb{E}_V(x)$ along with true function and GP regression, top figure; $\mathbb{E}_V(x)$, bottom figure. The constraint is $c_1(x) = y(x) - 2 \geq 0$. Shaded area denotes an uncertainty region of $\pm 2s$.

the value of constraint j at the midpoint of rectangle r. In addition, let $\mu_1, \ldots, \mu_m$ be positive weighting coefficients for the inequality constraints. Finally, assume that the constrained global minimum value y^* is known. The nonnegative auxiliary function, evaluated at the center of rectangle r, is:

$$\phi(x^{(r)}; \mu, y^*) = \max(y(x^{(r)}) - y^*, 0) + \sum_{j=1}^{m} \mu_j \max(c_j(x^{(r)}), 0) \tag{4.8}$$

This is not a penalty function in the standard sense. For the global minimum to occur in rectangle r, the auxiliary function must fall to zero starting from its value $\phi(x^{(r)}; \mu, y^*)$ at the center point. Moreover, the maximum distance over which this change can occur is the center-vertex distance d_r. Thus, to reach the global minimum in rectangle r, the auxiliary function must undergo a minimum rate of change, denoted $h_r(y^*)$, given by

$$h_r(y^*) = \frac{\phi(x^{(r)}; \mu, y^*)}{d_r} \tag{4.9}$$

Since it is more reasonable to expect gradual changes than abrupt ones [55], a reasonable way to select a rectangle would be to select rectangles that minimize the rate of change $h_r(y^*)$. Of course, this is impractical because y^* is generally unknown, but it is possible to select the set of rectangles that minimize $h_r(y^*)$ for some $y^* \leq y_{\min} - \epsilon$. This assumes that a feasible point has been found. If no feasible points have been found, the rectangles chosen are such that $h_r(y^* = 0)$ is minimized, i.e., the rectangles are chosen where the weighted constraint violations can be brought to zero with the least rate of change. In the unconstrained case, the strategy reduces to that in §3.4.2.

For this research, `DIRECT` has been modified to also handle hard constraints. Assuming at least one successful (but not necessarily feasible) point exists so far, a failed value that is returned at some point $x^{(f)}, f \in \mathcal{F}$ is assigned the value

$$y(x^{(f)}) \leftarrow \max_{\substack{1 \leq i \leq k \\ i \notin \mathcal{F}}} y(x^{(i)}) \tag{4.10}$$

Inequality constraints, if any, are similarly assigned as

$$c_j(x^{(f)}) \leftarrow \min_{\substack{1 \leq i \leq k \\ i \notin \mathcal{F}}} c_j(x^{(i)}), \qquad j = 1, \ldots, m \tag{4.11}$$

This drives the search away from the infeasible region. In the case where no successful points have been returned, that point is assigned a very large positive value, say, 10^{30}, and the inequality constraints are assigned a very large negative value, say, -10^{30}. Until a successful point is returned, new points are chosen by selecting rectangles with the largest center-vertex distance, i.e., the search is purely global. The reader is referred to the references for additional details [53, 55].

4.4 Other Methods

The method of handling nonlinear constraints by setting the ISC to some undesirable value outside the feasible region has been proposed in the literature [110, 79]. Sobester et al. [110] propose a simple modification to their weighted expected improvement function $\mathbb{E}_I(x; w)$. The criterion is set to zero wherever the approximate constraints are violated:

$$\mathbb{E}_I(x; w) = \begin{cases} w(y_{\min} - \hat{y})\Phi(u) + (1-w)s\phi(u), & s > 0,\ \hat{C}_i \geq 0, i = 1, \ldots, m \\ 0, & \text{otherwise} \end{cases} \tag{4.12}$$

$y_{\min}$ is taken as the minimum feasible objective value, where feasibility is assessed on the basis of the approximate constraints. Osborne et al. [79] mention that the expected minimum criterion (a Bayesian ISC closely related to the EI criterion) can be extended to nonlinearly constrained problems by setting EM to $+\infty$ at infeasible locations. The method of constraint handling by setting the ISC to some value outside the feasible region does not account for the uncertainty in the constraints.

Besides the methods discussed above, there are few local methods that can be used for black-box optimization. A surface effect ship design problem [61] was originally solved using SUMT. The augmented Lagrangian method was used to convert the constrained problem to an unconstrained problem and the method of conjugate directions [84, 117] was used to minimize the unconstrained problem. A derivative-free method had to be used because of the technical challenges. The method of conjugate directions [84] is perhaps the most efficient of all existing direct local methods that have proven convergence properties, but its use within a SUMT and multistart framework may be prohibitively expensive.

M.J.D. Powell has developed a family of derivative-free trust region algorithms [85, 87, 86, 88, 89] for use on problems that are subject to some of the technical challenges. The algorithms do not haven proven convergence properties but they work well in practice. Only the algorithm Constrained Optimization BY Linear Approximation (COBYLA) [85] is able to directly handle nonlinear constraints. COBYLA constructs successive linear polynomial approximations to the objective and constraint functions by interpolation at the vertices of a simplex of $n + 1$ points and minimizes the approximations within a trust region at each step.

CHAPTER V

ADDITIONAL CONSIDERATIONS

5.1 Experimental Designs

The goal of surrogate model-based global optimization is to focus resources in promising areas where the global minimum is likely to occur. The influence of experimental designs was found to have a significant effect on the performance of expensive black-box global optimization algorithms [91]. In this research, the influence of experimental designs is investigated to a greater degree. The remaining designs are standard designs that only vary by dimensions; there is no stochastic component in the initial design thus the designs do not contribute any variability to the algorithm performance. The DGS design is problem-specific but has no stochastic component. Three distinct strategies plus two combination strategies are considered:

1. **Deterministic global solver (DGS)** A black-box global optimizer such as `DIRECT` [53] is utilized for a limited number of function evaluations and the results are used as the initial set of samples for the surrogate model-based algorithm. The samples chosen by `DIRECT` are sequential and only the first three points are systematically placed; the remaining points are problem specific.

2. **Corner point strategy (CPS)** This strategy has a fixed number of samples. For problems with $n \leq 3$, the midpoint plus all corners of the bounding box plus all corners of the half-bounding box $\{x | \frac{1}{4} \leq x \leq \frac{3}{4}\}$ on the unit hypercube are taken, giving $2^{n+1} + 1$ samples. For problems with $n > 3$, a fractional corner point design (FCP) is implemented to avoid the curse of dimensionality. This is described next. Assume that the bounding box A has been transformed to the unit hypercube. Then the lower left corner at $(0, 0, \ldots, 0)$ plus the upper

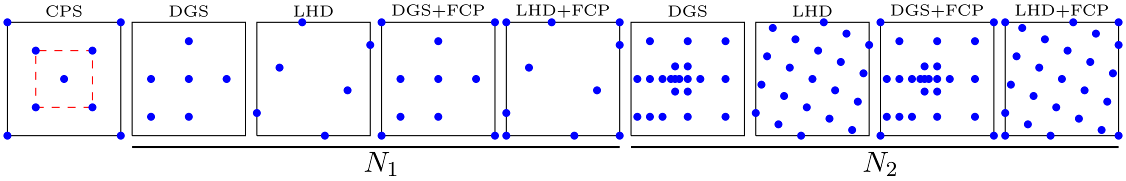


Figure 38: The nine designs from Table 5 illustrated in two dimensions

right corner at $(1, 1, \ldots, 1)$ plus all corner points adjacent to $(0, 0, \ldots, 0)$ and $(1, 1, \ldots, 1)$ are sampled, giving $2(n+1)$ samples. In two and three dimensions, FCP is equivalent to sampling all corner points.

3. **Maximin LHD** Latin hypercube designs are a popular choice for experimental designs because they evenly sample the design space. Maximin designs attempt to maximize the minimum distance between samples, thus ensuring an even spread. The maximin LHDs used in this research are optimal designs obtained from [116].

4. **Combined designs** Combinations of the FCP design with either DGS or maximin LHD were also tested.

For all designs except CPS, two different numbers of initial points, which are commonly used in the literature, are considered: $N_1 = (n+1)(n+2)/2$ and $N_2 = 10n + 1$ [91], with N_1 being the number of points required to fit an interpolating quadratic polynomial to the data and N_2 being a “rule of thumb” from the literature for the size of space-filling designs. The designs are summarized in Table 5 and illustrated in Figure 38 in two dimensions.

5.2 Initially Infeasible Designs

If the initial sample does not contain a feasible point, a “Phase I”-type approach can be employed [110]. In classical optimization this can be accomplished by applying a minimization algorithm to the constraint violations instead of the original objective

Table 5: Experimental designs. Five designs are listed with available options and total combinations.

Acronym	Design	Size	i
CPS	Corner point strategy	$\begin{cases} 2^{n+1}+1, & n \leq 3 \\ 2(n+1), & n > 3 \end{cases}$	–
DGS	Deterministic global solver	N_i	1,2
LHD	Maximin LHD	N_i	1,2
DGS+FCP	DGS + Fractional Corner Points	$N_i + \min(2^n, 2(n+1))$	1,2
LHD+FCP	LHD + Fractional Corner Points	$N_i + \min(2^n, 2(n+1))$	1,2

function, i.e.,

$$\underset{x\in\mathbb{R}^n}{\text{minimize}} \sum_{i\in\mathcal{E}} |c_i(x)| + \sum_{i\in\mathcal{I}} (-c_i(x))_+ \tag{5.1}$$

Once a feasible point has been found, the the original problem can be solved. In Bayesian optimization the probability of feasibility can be maximized [105]:

$$\underset{x\in\mathbb{R}^n}{\text{maximize}} \quad \prod_{i=1}^{m} \mathrm{P}\left(C_i \geq 0\right) \tag{5.2}$$

Once a feasible point has been found, an improvement-based ISC can be used for subsequent iterations. Informal numerical experiments indicate that Phase I is not required for small problems, as a Bayesian algorithm will automatically search for feasible points due to the form of equation (4.4). However, it may help to locate feasible points for larger problems in fewer function evaluations than the constrained EI criterion alone.

5.3 Missing Data

One of the technical challenges outlined in §1.2 was the existence of hard or hidden constraints which may return non-numerical values, e.g., `NaN` or `Inf`, or may cause the black-box routine to exit at approximately the same cost as a successful iteration. Neither event poses any difficulty provided that a value for the objective and constraint functions is available by some other method. Furthermore, for all the algorithms discussed, future iterates depend only upon past data unlike, e.g., quasi-Newton methods which maintain updates throughout iterations. An important distinction [39] for missing data is whether or not the data is missing at random. If the data is missing at random it can be ignored, as in the case of an initial sample set generated by a space-filling design. When the data is missing due to a hard constraint and is not missing at random, it is necessary to assign a value to the missing data. One strategy [39] is to assign to the failed point $x^{(f)}, f \in \mathcal{F}$ the value $y(x^{(f)}) \leftarrow \hat{Y}(x^{(f)}) + s^2(x^{(f)})$, i.e., to assume its true value is equal to the mean value $\hat{Y}(x^{(f)})$ predicted by the model but penalized by a statistical upper bound $s^2(x^{(f)})$. After each iteration, a GP model is constructed based on the feasible iterates only and used to impute the values of the failed past iterations. The goal is to drive the search away from this region, which is accomplished in two ways: by reducing the uncertainty around the failed point and by assigning it a large function value.

In the current research, the penalty of $s^2(\cdot)$ is changed to $s(\cdot)$ to provide better control for the magnitude of the penalty. The change prevents the penalty $s^2(\cdot)$ from becoming too large when $s(\cdot) > 1$, as this may warp the GP model. Additionally, the following strategy is proposed for the constraint values. By (NLP), a design is feasible if $c_i(x) \geq 0$, $i = 1, \ldots, m$. Thus for $x^{(f)}$, the constraint value of $c_i(x^{(f)}) \leftarrow \hat{C}_i(x^{(f)}) - s_i(x^{(f)})$. This will drive the search away from the neighborhood of $x^{(f)}$ due to infeasibility.

The expressions for $y(x^{(f)})$ and $c_i(x^{(f)})$ in the above two paragraphs preserve the global converge properties of the algorithm. Far from sample sites, i.e., as $\|x - x^{(i)}\|_2 \to \infty$, the predictor approaches the mean value of the iterates, i.e., $\hat{Y}(x) \to \hat{\beta}$, but the penalized prediction approaches $\hat{\beta} + s(x)$. Close to sample sites, i.e., as $\|x - x^{(i)}\|_2 \to 0$, the predictor approaches the observed value, i.e., $\hat{y}(x) \to y(x^{(i)})$, which is necessary for maintaining the asymptotic convergence of improvement-based algorithms. Similar arguments can be made for the constraint penalization strategy.

Additional strategies which are considered in §9.1 are predictor imputation, i.e., $y(x) \leftarrow \hat{Y}(x^{(f)})$, imputation by maximum (successful but not necessarily feasible) value, i.e., $y(x) \leftarrow \max\limits_{\substack{1 \leq i \leq k \\ i \notin \mathcal{F}}} y(x^{(i)})$, and a random update strategy.

5.4 Sampling In The Presence Of Noise

If $\mathcal{S}$ is corrupted by output-dependent noise such that the observed values are now $z(x) = y(x) + \epsilon$, a noise term $\epsilon_n = \delta_{ii}\theta_n$ can be appended to the covariance function, where δ_{ii} is the Kronecker delta and θ_n is the noise variance which is treated as an additional hyperparameter. The GP will no longer interpolate the observations; rather, it will regress them. It is well known that noise corrupted data should be regressed rather than interpolated because the regression acts as a noise filter. The situation is illustrated in Figure 39, which shows the drag polar of a NACA 2432 airfoil. The polar has been calculated using the panel code XFOIL 6.94 [32, 31]. The numerical simulation is subject to discretization error and incomplete convergence which leads to noise corrupted results, causing the interpolative GP to behave erratically; the regression gives a more reasonable fit.

In deterministic experiments, artificial noise may also be added to improve the conditioning of $\mathbf{K}$. For example, values of θ_n on the order of 10^{-6} have been used in [57]. In this research, a more stringent condition is applied when selecting values of θ_n. The value of θ_n is chosen such that $\mathcal{O}(\theta_n / \max\{\theta_i : i = 1, 2, \ldots, n\}) \approx 10^{-8}$. For isotropic covariance functions, this condition reduces to $\mathcal{O}(\theta_n/\theta) \approx 10^{-8}$. This relation has been observed to work well in numerical experiments and is expressed in this relative manner to ensure that the when the observations $y^{(k)} \ll 1$, the GP model is not polluted by artificial noise. It is recommended that $y^{(k)}$ be scaled to $\mathcal{O}(1)$ in order to keep the noise level at approximately the the square root of machine

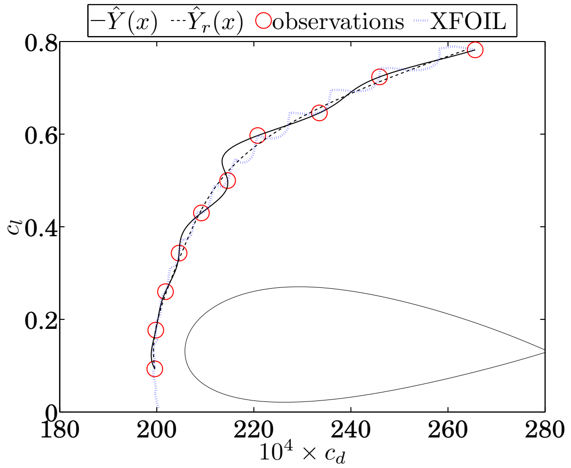


Figure 39: Drag polar from XFOIL 6.94 for NACA 2432 airfoil discretized into 280 panels at $M = 0.100$ and $Re = 0.200 \cdot 10^6$; observations (open circles), GP mean interpolator (solid line), GP mean regressor (dashed line), high-resolution XFOIL calculation (wavy dotted line). Airfoil shown in 1:1 aspect ratio.

epsilon for double precision.

Because a GP regression will no longer interpolate the observations, there will be a nonzero error at these sites. This leads to the possibility of resampling at previously sampled locations, which for deterministic experiments results in no new information and may cause an algorithm to stall. Forrester et al. [38] introduce the concept of reinterpolation to prevent resampling. Although there is nonzero error in all sample locations due to noise, because the observations are deterministic, the notion of error can be redefined to reflect the uncertainty in the result. Zero error at the sample locations is achieved by constructing a secondary interpolating GP through the values predicted by the GP regression at the sample locations. Intuitively, the mean function of the reinterpolation will be identical to the mean function of the regression GP. This is shown first. Following the development in §2.2.5, the predictor of the reinterpolation is

$$\mu(x) = \hat{\beta} + \mathbf{r}_0^T \mathbf{R}^{-1}(y_r^{(k)} - \hat{\beta}) \tag{5.3}$$

where

$$\hat{\beta} = \frac{1^T \mathbf{R}^{-1} y_r^{(k)}}{1^T \mathbf{R}^{-1} 1} \tag{5.4}$$

The values $y_r^{(k)}$ are obtained from the regression mean function as

$$y_r^{(k)} = \hat{\beta}_r + \mathbf{R}(\mathbf{R} + \mathbf{I}\theta_n)^{-1}(y^{(k)} - \hat{\beta}_r) \tag{5.5}$$

Substituting equation (5.5) into equation (5.4) gives

$$\begin{aligned}
\hat{\beta} &= \frac{1^T \mathbf{R}^{-1} \hat{\beta}_r + 1^T \mathbf{R}(\mathbf{R} + \mathbf{I}\theta_n)^{-1}(y^{(k)} - \hat{\beta}_r)}{1^T \mathbf{R}^{-1} 1} \\
&= \hat{\beta}_r + \frac{1^T (\mathbf{R} + \mathbf{I}\theta_n)^{-1} y^{(k)} - 1^T (\mathbf{R} + \mathbf{I}\theta_n)^{-1} \hat{\beta}_r)}{1^T \mathbf{R}^{-1} 1} \\
&= \hat{\beta}_r
\end{aligned}$$

where the last expression above is obtained by noting that $1^T(\mathbf{R} + \mathbf{I}\theta_n)^{-1} y^{(k)} = 1^T(\mathbf{R} + \mathbf{I}\theta_n)^{-1}\hat{\beta}_r$. Now, substituting equation (5.5) into equation (5.3) and replacing $\hat{\beta}$ with $\hat{\beta}_r$ yields

$$\begin{aligned}
\mu(x) &= \hat{\beta}_r + \mathbf{r}_0^T \mathbf{R}^{-1}(\hat{\beta}_r + \mathbf{R}(\mathbf{R} + \mathbf{I}\theta_n)^{-1}(y^{(k)} - \hat{\beta}_r) - \hat{\beta}_r) \\
&= \hat{\beta}_r + \mathbf{r}_0^T (\mathbf{R} + \mathbf{I}\theta_n)^{-1}(y^{(k)} - \hat{\beta}_r) \\
&= \mu_r(x)
\end{aligned}$$

Thus, the regression mean function and the reinterpolation mean function are identical and the mean of the regression model may be used as the reinterpolation mean. The variance of the predictive distribution (2.26) must now be updated to reflect the condition that the reinterpolation mean interpolates $y_r^{(k)}$. Replacing $y^{(k)}$ in equation (2.25b) with $y_r^{(k)}$ from equation (5.5) yields, after some basic algebra,

$$b_{k,r} = b_0 + \frac{1}{2}\left[(y^{(k)})^T(\mathbf{R} + \mathbf{I}\theta_n)^{-1}\mathbf{R}(\mathbf{R} + \mathbf{I}\theta_n)^{-1} y^{(k)} - \left(1^T(\mathbf{R} + \mathbf{I}\theta_n)^{-1} y^{(k)}\right)^T \hat{\beta}\right] \tag{5.6}$$

The new predictive distribution which now reports zero error at sample sites is

$$p(y_0|y^{(k)}, \theta) \sim t_{\eta_k = 2a_k}\left(\mu(x), \gamma_k^2(x) \triangleq (b_{k,r}/a_k)\kappa^2(x)\right) \tag{5.7}$$

The advantage of this approach is that only the error needs to be updated; there is no need to recompute the hyperparameters θ and the regression mean $\mu(x)$ can still be used as the reinterpolation mean. The methods are employed within a fully Bayesian GP model and illustrated in Figure 40 for the NACA 2432 airfoil shown in Figure 39. Notice that the mean function is the same for both methods, but the error for the reinterpolation returns to zero at all the observation sites. The expected improvement (not shown) will also return to zero for all observation sites, thus an update strategy based on reinterpolation eliminates the possibility of resampling and preserves the convergence properties of the algorithm.

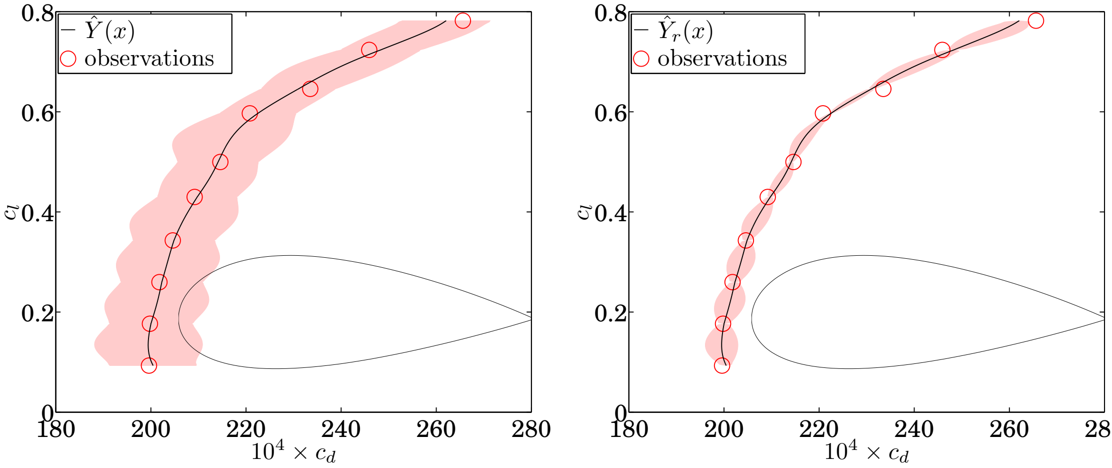


Figure 40: Fully Bayesian GP regression without reinterpolation, left; GP regression and error with reinterpolation, right. Shaded area denotes an uncertainty region of $\pm 2s$.

Example 5.4.1 Fully Bayesian regression of noise corrupted observations

There is no difficulty in applying the fully Bayesian approach to problems with noise corrupted observations. A noise hyperparameter θ_n is added as discussed in §2.2 to the prior $p(\theta)$ and the predictive distribution (2.29) is computed as usual. This procedure is illustrated on the airfoil problem from Figure 39. Figure 41 shows the airfoil data, the fully Bayesian interpolation, and the fully Bayesian regression. Figure 42 shows the posterior $p(\theta, \theta_n | y^{(k)})$ for a uniform prior $p(\theta, \theta_n)$.

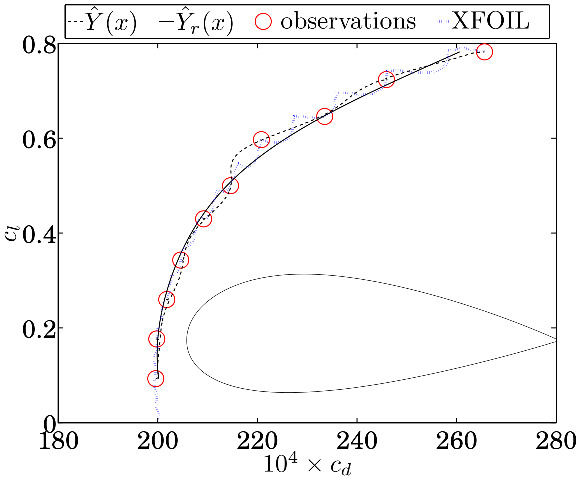


Figure 41: Regression and interpolation of drag polar data from XFOIL 6.94 for NACA 2432 airfoil using a fully Bayesian approach. Airfoil shown in proper 1:1 aspect ratio.

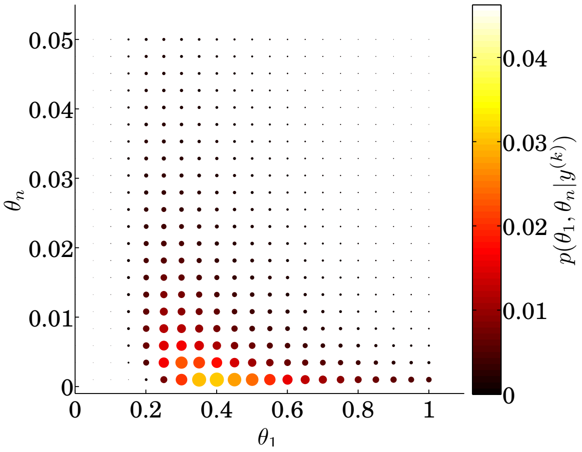


Figure 42: Posterior distribution $p(\theta, \theta_n | y^{(k)})$ for airfoil problem.

□

5.5 Stopping Rules

surrogate model-based algorithms do not necessarily make the same assumptions about the unknown function $y(\cdot)$ as do convergent local optimization algorithms. In general, different stopping rules are used to terminate surrogate model-based optimization algorithms.

For bound constrained problems, algorithms will typically terminate when the current best iterate $y_{\min}$ comes within some prescribed tolerance ϵ of the global minimum value, if this value is known. The conditions (8.3) implement this rule. More advanced stopping rules for this situation have also been used. The P-algorithm, for example, terminates when the probability of the evaluation of the global minimum with the given accuracy exceeds some value close to 1 [121, 122]. When the global minimum value is unknown, other rules must be used which may be particular to each algorithm. For example, EGO terminates when the *relative* EI is within a user-prescribed ϵ of the current best function value. The advantage of using this probabilistic stopping rule is that the true minimum of the function does not need to be known and often it will remain unknown. However, it was discovered in the course of this research that stopping rules based on the EI criterion can only be used with interpolation problems. For these problems, the samples serve as control points where the EI returns to zero and gradually diminishes as more samples are added. For noise corrupted problems, there are no control points to ensure such behavior. Even with reinterpolation, the set of reinterpolated values changes from iteration to iteration, which changes EI. superEGO [105] depends primarily on a function evaluation limit for termination, but also terminates if N sample points have been generated within a certain distance of each other. However, this second stopping rule may cause the algorithm to fail if used with a sampling criterion that tends to search locally in early iterations. Other stopping criteria include goal attainment, i.e., stopping when a certain function value has been reached, lower confidence bounding [54, 27], i.e., stopping when the lower uncertainty is less than some desired percentage of the posterior variance, and stopping when the cost to evaluate the next sample outweighs the benefits [13]. All algorithms also employ a function evaluation limit as a secondary stopping criterion when the primary criterion is not met.

For nonlinearly constrained problems, stopping rules in the literature [10] are typically based on a function evaluation limit. Some algorithms, e.g., superEGO, terminate after some user-prescribed number N of feasible points have been found. There is a certain degree of arbitrariness required when developing stopping rules due to the level of constraint violation one is willing to accept in the inequality constraints. While inequality constraints may always be satisfied for some problems, equality constraints may never be satisfied to machine accuracy, thus one must accept a violation or allow a relaxation factor as in equation (1.2).

CHAPTER VI

METHODOLOGY FOR GLOBAL OPTIMIZATION OF COMPUTATIONALLY EXPENSIVE DESIGN PROBLEMS

6.1 fBcEGO: Fully Bayesian Constrained Efficient Global Optimization

The research thus far is used to develop the specification of a fully Bayesian constrained efficient global optimization algorithm, henceforth called fBcEGO. A matrix of alternatives for the algorithm components is populated and used to downselect the components (Table 6). The literature review and extensive analysis of the existing methods thus far has led to the elimination of a large number of alternatives, which have been grayed out in Table 6. For the initial designs, smaller designs are preferred because they will allow fBcEGO to start placing samples sooner. Factorial and random designs are eliminated due to their large size, and fractional corner point and DGS designs are retained due to their small size and utility in expensive optimization. Maximin LHDs will be tested to obtain a conclusive decision on their utility, as these designs tend to be large, but lead to globally accurate surrogate models [92], which may benefit the global optimization algorithm. The surrogate model type was selected in Chapter 2. For the ISC, the expected improvement was selected for its use with Gaussian process models and the automatic balance that it provides between the global and local search. DIRECT was found to be the most efficient ISC subsolver because it is a global method that only uses function evaluations. This reduces the precision-related errors that may arise in the maximization problem versus finite difference gradient-based methods when EI becomes small. Branch & bound methods are not applicable to the solution of the fully Bayesian EI criterion and multistart and evolutionary algorithms require many more function evaluations than DIRECT to obtain similar performance. For constraint handling, the constrained EI criterion was shown to be a powerful method for handling the nonlinear constraints. Finally, the penalized imputation method was selected to address the hard constraints, but the predictor and maximum value imputation methods will be tested for comparison.

`fBcEGO` is described in the remainder of this section and a formal algorithm specification is outlined in §6.2. A step-by-step methodology is proposed in §6.3 which may help a user employ `fBcEGO` to solve expensive black-box design problems.

Table 6: Matrix of alternatives for algorithm components.

Component	Alt. 1	Alt. 2	Alt. 3	Alt. 4	Alt. 5
Initial Design	Fractional corner point	DGS	Maximin LHD	Factorial	Random
Termination	Relative or absolute change	Based on ISC	Based on surrogate model	Function evaluations	Statistical
Surrogate model	Gaussian process	Radial basis function	Polynomial response surface	Neural network	Support vector regression
ISC	Expected minimum	Probability of improvement	Expected improvement	Minimize model	Maximize smoothness
ISC Subsolver	Branch & bound	Gradient-based multistart	Derivative-free multistart	DIRECT	Evolutionary
Constraint handling	SQP	Interior-point	Filter	Penalty	Probabilistic criterion
Noise	Regression	Reinterpolation			
Hard constraints	Random update	Predictor imputation	Penalized imputation	Max. value imputation	

The method for selecting hyperparameter priors was outlined and demonstrated in §2.2.6. The remainder of this section outlines a global optimization algorithm that is based around this method and also employs some additional techniques derived from the literature to address all the technical challenges from §1.2. The algorithm is iterative and exactly one point is selected per iteration. This point will serve as the next observation site to be evaluated by the black-box function.

The algorithm begins by evaluating N initial samples from an experimental design, which will result in N values for the objective function and each of m constraints, giving $N(m+1)$ initial values. It is not necessary for all values to be feasible; a feasibility phase is optional but recommended for larger problems. However, the number of successful initial samples must satisfy condition (2.28).

The next step is to construct the GP models for the objective and constraints using the method described in §2.2.3. Recall that a strategy to select the prior $p(\theta)$ was outlined in this section. Hard constraints must also be addressed at this stage using one of the penalization methods described in §5.3. The next iterate is selected by maximizing the constrained EI. A gradient-free method is required since it was not possible to derive an asymptotic expansion to the fully Bayesian constrained EI (see Remark 3.2.3). `DIRECT` is preferred for this purpose. The best maximizer of EI found in this way is then refined using a local solver and evaluated by the black-box function to give the next observation. The sample sets are updated and the algorithm repeats in this manner until termination.

6.2 fBcEGO Specification

A description of the global optimization algorithm was given in the previous section. A detailed specification is outlined here.

1. **Initialize**

   1.1 Select initial design $\mathcal{D}^{(1)} = \{x^{(1)}, \dots, x^{(k)}\} \subset A$

   1.2 Compute the initial sample sets $\mathcal{S}^{(1)} = \{y(x^{(1)}), \dots, y(x^{(k)})\}$ and $\mathcal{C}_j^{(1)} = \{c_j(x^{(1)}), \dots, c_j(x^{(k)})\}$ for $j = 1, \dots, m$

   1.3 Select initial priors $p(\theta)$ for Y and C_j for $j = 1, \dots, m$

   1.4 Set $i \leftarrow 1$

2. **Iteration step:** While termination condition is not satisfied do:

   2.1 **Impute failed values:** If any points in $\mathcal{D}^{(i)}$ have returned failed values for $y(\cdot)$ or $c_j(\cdot)$, construct a GP through the successful sites $\mathcal{D} \setminus \{x^{(f)}\}, f \in \mathcal{F}$ and impute values $y(x^{(f)}) \leftarrow \hat{Y}(x^{(f)}) + s(x^{(f)})$ and $c_j(x^{(f)}) \leftarrow \hat{C}_j(x^{(f)}) - s_j(x^{(f)})$ for $j = 1, \ldots, m$, respectively, for each failed point $x^{(f)}$

   2.2 **Construct models for the objective and constraints**

   2.2.1 **Update priors:** Compute $p(\theta|y^{(k)})$ and $p(\theta|c_j^{(k)})$ for $j = 1, \ldots, m$ and update $p(\theta)$ for the objective and constraints using the strategy outlined §2.2.3

   2.2.2 **Select modes:** Sort $p(\theta|y^{(k)})$ and $p(\theta|c_j^{(k)})$ by descending order and continue with the θ that correspond to the top 99.99% of the mass of $p(\theta|y^{(k)})$ and $p(\theta|c_j^{(k)})$, respectively

   2.2.3 **Fit models:** Compute the predictive distributions $p(Y|y^{(k)})$ and $p(C_j|y^{(k)})$ for $j = 1, \ldots, m$, i.e., fit GP models to $\mathcal{S}^{(i)}$ and $\mathcal{C}_j^{(i)}$ for $j = 1, \ldots, m$ through $\mathcal{D}^{(i)}$

   2.3 **Compute next iterate:** Compute the next sample point by solving

   $$x^{(k+1)} = \arg\max_{x \in A} \mathbb{E}_\theta \left[\mathbb{E}_{I|y^{(k)},\theta}(x)\right] \mathbb{E}_\theta \left[\Pi_{j=1}^m \mathrm{P}\left(C_j \geq 0|\theta, y^{(k)}\right)\right]$$

   2.4 **Perform expensive evaluation:** Perform expensive evaluation $y(x^{(k+1)})$

   2.5 **Update:** $\mathcal{D}^{(i+1)} \leftarrow \mathcal{D}^{(i)} \cup x^{(k+1)}$, $\mathcal{S}^{(i+1)} \leftarrow \mathcal{S}^{(i)} \cup y(x^{(k+1)})$, $\mathcal{C}_j^{(i+1)} \leftarrow \mathcal{C}_j^{(i)} \cup c_j(x^{(k+1)})$ for $j = 1, \ldots, m$, $i \leftarrow i+1$, $k \leftarrow k+1$, $y_{\min} \leftarrow \min_{x \in A, 1 \leq i \leq k} y(x^{(i)})$, $\{i : i \notin \mathcal{F}, c_j(x^{(i)}) \geq 0, 1 \leq j \leq m, 1 \leq i \leq k\}$

6.3 Methodology

The methodology for global optimization of computationally expensive design problems is outlined in this section. A flowchart of the complete methodology including fBcEGO is depicted in Figure 43.

6.3.1 Step 1: Formal Problem Statement

The first step in the methodology is to provide a formal problem statement for fBcEGO. This requires identification of the following information: the design variables x and their lower and upper bounds x^ℓ and x^u, respectively; the objective function $y(\cdot)$; the equality and inequality constraints $c_i(\cdot)$, $i \in \mathcal{E}$ and $i \in \mathcal{I}$, respectively. If bounds on x are unknown, they should be determined using some method that does not require evaluation of the expensive function $y(\cdot)$. For instance, in a geometry design problem, the space may be bounded by eliminating regions corresponding to nonsensical geometries. In other cases, bounds may be defined by customer requirements. If there is no inexpensive method to determine the bounds, then small space-filling designs may be used to explore A. These evaluations can then be used in the initial design $\mathcal{D}^{(1)}$ and no function evaluations are wasted.

The black-box design problem must be linked to fBcEGO. Typically, a design program will read inputs from a file or the command line, and write outputs to the screen or a file. A wrapper may be required in order to create an accessible method. These implementation details are left to the user. Once the required information is collected, the problem is described in the form:

$$\begin{aligned} \underset{x\in\mathbb{R}^n}{\text{minimize}} \quad & y(x) \\ \text{subject to} \quad & c_i(x) = 0, \qquad i \in \mathcal{E} \\ & c_i(x) \geq 0, \qquad i \in \mathcal{I} \\ & x^\ell \leq x \leq x^u \end{aligned} \tag{6.1}$$

This problem is explicitly written with the equality constraints because fBcEGO internally converts these to two inequality constraints.

6.3.2 Step 2: Select Initial Design

Next, an initial design must be supplied. Some recommendations are made based on the results in Chapter 8.2. In general, smaller initial designs result in better performance because the algorithm is allowed to intelligently place samples sooner. Any function evaluations made in the past can be included in the initial design.

6.3.3 Step 3: Select Stopping Criteria

Stopping criteria were discussed in §5.5 and are automatically determined based on the information given to the algorithm, but the user must always supply a function evaluation budget. In some cases, this may be the only stopping criterion.

6.3.4 Step 4: Select Covariance Function

The last piece of information required before the algorithm can execute is the covariance function. Some recommendations are given in §8.6 based on the level of information known about the problem. In general, if nothing is known about the problem or if it is known that the problem is mildly nonsmooth, the isotropic SE covariance function should be used. Other options exist to encapsulate prior information. A poor choice for the covariance function can degrade performance.

6.3.5 Step 5: Execute `fBcEGO`

From an implementation perspective, the minimal inputs to `fBcEGO` are the black-box function which takes x as the input and outputs $y(x)$ and $c_j(x), j = 1, \ldots, m$, the initial design $x^{(k)}$, the number of equality constraints, the number of inequality constraints, and the bounds of x. Stopping criteria, covariance functions, and subsolvers for the ISC maximization may also be specified; otherwise, `fBcEGO` uses some default options.

With this information in hand, `fBcEGO` can be executed. It is assumed that the user has no knowledge about the hyperparameters for the design problem and thus does not need to modify the assumptions placed on the priors. The method for selecting the hyperparameter priors builds them automatically. The outputs of the algorithm are x^*, y^*, $x^{(k)}$, $y^{(k)}$, $c_j^{(k)}$ for $j = 1, \ldots, m$, the equality constraint violation history, the indices of the failed iterations, and a structure that encapsulates the GP model for visualization.

6.3.6 Step 6: Visualization & Analysis

When `fBcEGO` terminates, the design space may be visualized through a number of methods available in the literature [56, 77, 50]. These may be application specific and details are left to the user. In general, because the goal of surrogate model-based global optimization algorithms is to focus resources in promising areas, the model will be most reliable in minima where the samples are concentrated. In practice however, after a large number of function evaluations, `fBcEGO` will have explored the design space in a space-filling manner due to the contribution of the uncertainty $s(\cdot)$ to the EI criterion.

An important part of this step is for the user to determine if `fBcEGO` needs to be executed again. For example, the best solution that was found may have been on the boundary of the design space, in which case the user may wish to return to Step 1 to expand the bounding box and continue the search for better solutions. This does not pose a problem for `fBcEGO` since surrogate model-based methods only depend on function evaluations made in the past. This is in contrast to, e.g., quasi-Newton methods which maintain updates of the Hessian matrix that are lost if the algorithm is restarted. Because of the dependence on past function evaluations, the user may return to any step of the methodology to make changes. The covariance function can be changed, the stopping criteria can be changed, and for some design problems, if Step 1 is written correctly, the constraints can be relaxed or a new problem can be written altogether without losing any previous work. Design problems that have this property are those that report values of some performance parameter rather than values for the constraint violation. The constraint function is then constructed using this value, e.g., if the value of some response $f(x)$ is reported by the design code, a normalized constraint $c(x)$ with an upper limit f^u can be written as

$$c(x) = 1 - \frac{f(x)}{f^u} \geq 0 \tag{6.2}$$

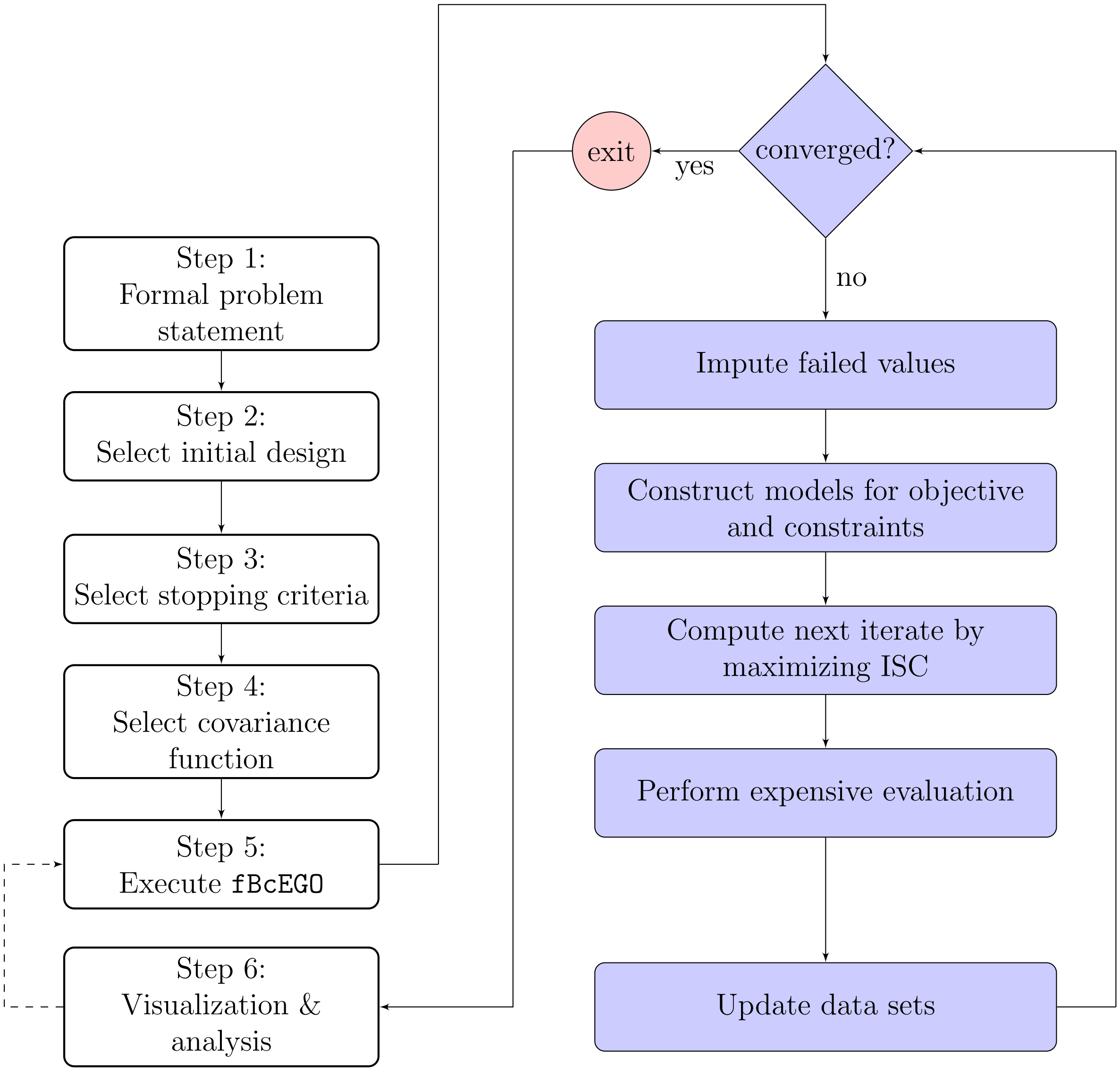


Figure 43: Methodology flowchart. Steps of the methodology (left) and `fBcEGO` (right). The iterative step from Step 6 to Step 5 is optional.

CHAPTER VII

HYPOTHESIS & TESTING PLAN

7.1 Research Questions & Hypotheses

The primary research objective was to solve problem (NLP) subject to the technical challenges in §1.2. The primary research question was posed in §1.3. This section presents the primary hypothesis of this research. A set of low-level research questions regarding specific, isolated aspects of `fBcEGO` are posed, which are refined versions of the questions in §1.3. Subhypotheses are presented as potential answers to each low-level research question. Test problems, competing algorithms, and test metrics are described, which are used to design experiments in §7.6 that will be used to evaluate each subhypothesis.

Primary Hypothesis

Within the context of surrogate model-based global optimization, an algorithm employing fully Bayesian GPs to model the objective and constraint functions and using a fully Bayesian constrained EI criterion as the ISC will solve a larger percentage of NLP-type problems in fewer function evaluations than the state-of-the-art methods found in the literature.

The hypotheses were derived based on the observations thus far. The first hypothesis deals with initial designs in surrogate model-based global optimization algorithms. The second hypothesis deals with the performance ISCs on bound constrained problems, i.e., when no nonlinear constraints are present. The third hypothesis deals with constraint handling techniques for nonlinear constraints. The fourth hypothesis deals with noise corrupted observations. The fifth hypothesis deals with nondifferentiabilities that may arise in design problems and how a surrogate model can address these. The sixth hypothesis deals with imputation methods for assigning values to failed iterations.

Research Question 1: How does the initial design affect the performance of algorithms in terms of number of simplex gradients $N_{p,s}$ required to solve problems to within some accuracy ϵ?

Hypothesis 1: Smaller initial designs with points placed by `DIRECT` and augmented with a fractional corner point design, i.e., design that includes only some of the corners, will enable algorithms to solve more problems in fewer median simplex gradients than the same algorithms employing other systematic or nonsystematic initial designs.

Research Question 2: Within the context of GP-based global optimization, what ISC has the highest potential to obtain the largest reduction in the function values of computationally expensive black-box problems under budget constraints? How does the performance of non-GP-based algorithms compare?

Hypothesis 2: An algorithm employing the fully Bayesian EI criterion will solve more bound constrained problems than any other state-of-the-art algorithm for bound constrained problems. The accuracy relative to y^* is preferred versus the reduction in function value as the test metric because the reduction must be measured from some datum value, which may mask the true performance of the algorithm if the datum value is large. Bayesian algorithms are expected to perform better than non-Bayesian algorithms.

Research Question 3: Within the context of GP-based global optimization, how should nonlinear constraints be handled such that the resulting algorithm will solve more problems to a higher degree of accuracy given a budget? How does the performance of non-GP-based constraint-handling methods compare?

Hypothesis 3: An algorithm employing a fully Bayesian constrained EI criterion will solve more nonlinearly constrained problems to a higher degree of accuracy given a budget than any other state-of-the-art algorithm. An algorithm based on this ISC will also attain the lowest equality constraint violation when the budget is exhausted. Bayesian constraint-handling techniques are expected to perform better than non-Bayesian techniques.

Research Question 4: What strategy or strategies can be used to handle observations which have been corrupted by deterministic noise?

Hypothesis 4: An algorithm that employs a fully Bayesian GP regression strategy to model noise corrupted functions and a fully Bayesian EI criterion to search them will solve more noise corrupted problems to a higher degree of accuracy under a given budget and noise level than any competing algorithm.

Research Question 5: If a problem is believed to be nonsmooth, i.e., exhibits non-differentiable subspaces, how can this belief be included in a surrogate model-based algorithm? What if this belief is incorrect?

Hypothesis 5: A fully Bayesian algorithm that employs an isotropic nonsmooth covariance function with a fully Bayesian EI criterion as the ISC with will solve more nonsmooth problems than any other state-of-the-art algorithm employing nonsmooth basis functions. Algorithms that employ smooth basis functions to solve smooth problems will outperform algorithms that employ nonsmooth basis functions for the same problems.

Research Question 6: When hard constraints are encountered, how should the failed values be imputed such that subsequent iterations are more likely to be successful?

Hypothesis 6: A fully Bayesian algorithm that employs a penalized imputation method to handle hard constraints will attain a lower function value for practical optimization problems with hard constraints under a given budget and also result in a larger number of successful iterations than the same algorithm employing predictor imputation or maximum value imputation.

7.2 Testing Plan

The following systematic plan, adopted from [9], will be used to study the behavior of algorithms and to evaluate the subhypotheses in the previous section:

1. Select test problems that address specific technical challenges:
   - Noise-free bound constrained problems
   - Noise-free nonlinearly constrained problems
   - Noise corrupted problems
   - Nonsmooth problems
2. Select competing algorithms for each problem set
3. Select test metrics
4. Create test matrices

5. Specify the experiments required to evaluate the hypotheses

6. Apply proposed algorithm to aircraft design problem

7.3 Test Problems

There is no generally accepted set of test problems for benchmarking optimization algorithms because no finite set is exhaustive. For local optimization, standard sets of test problems such as the `CUTE` collection [17] or the Hock-Schittkowski collection [49] have been used extensively in the literature. Global optimization test problem sets are often custom sets of problems selected from the literature; the most comprehensive collection to date is found in [37]. A set of test problems must be selected such that (a) no bias is induced towards a particular algorithm, (b) each problem remains within the scope of the algorithm, and (c) the set is as exhaustive as possible within the scope of the algorithm. Requirement (c) can be implemented by devising a classification scheme for optimization problems. Some candidate classifications can be found in [114, 37, 83], but these consist of a small number of categories totaling less than five and are not detailed enough.

The classification scheme in [49] is adapted for this research. The following string of letters and numbers is used to classify the test problems:

$$\text{OCD-K-s-n-}m_{\mathcal{E}}\text{-}m_{\mathcal{I}}\text{-b} \tag{7.1}$$

The classification string (7.1) takes the admissible letters and integers given in Table 7. The global optimization test problem set used for this research is compiled from [59, 36, 37]. The bound constrained and nonlinearly constrained test problems are summarized in Tables 8 and 9. These problems are chosen to emulate characteristics of design problems, i.e., nonlinearity, nonconvexity or unknown convexity properties, and for the nonlinearly constraint problems, at least one active nonlinear constraint was present. The test problem statistics for these sets are summarized graphically in Figures 44 – 46. In Figure 46 the independent variable is the equivalent number of inequality constraints, i.e., the sum of all inequality constraints plus twice the number of equality constraints. This is because an equality constraint is handled as two inequality constraints in the formulation of `fBcEGO`.

Remark 7.3.1 The use of inexpensive problems to test algorithms developed for expensive problems
The test problems are analytical problems with a negligible cost per function evaluation when compared with the analytical effort of one iteration of the competing algorithms (§7.4). However, this research is to develop an algorithm for expensive problems. Function evaluations are a natural metric of performance for benchmarking derivative-free algorithms when the analytical work of the algorithm is negligible compared with the cost of one function evaluation. Thus, for testing purposes, the computational time of the design problem does not factor into the performance and inexpensive functions can be used. A situation can be envisioned where an expensive function is tested and time is sped up during the evaluation period without having any effect on the results. □

The behavior of the algorithms on nonsmooth problems is also of interest, since it was stated in the technical challenges that some design problems may exhibit C^0 continuity. One way to derive nonsmooth problems is to modify the bound constrained problems such that $y_{\text{ps}}(x) = |y(x) + c| + \|Ax - b\|_1$, where c is a constant that shifts $y(x)$ such that its range contains both positive and negative values, A is a diagonal matrix of coefficients a_{ii} on x_i, $i = 1, \ldots, n$, and b is an $n \times 1$ column vector of constants. These problems are continuous but contain nondifferentiable subspaces. There is no guarantee that $|y(x) + c|$ has a unique minimizer, even if $y(x)$ has a unique minimizer. This was a challenge particular to the derivation of nonsmooth problems from their continuously-smooth counterparts. Nonsmooth problems were still derived from the bound constrained problems in Table 8, but the bound constrained problems required considerable modification through the term $\|Ax - b\|_1$ to ensure that only a finite number of global minimizers existed. The nonsmooth set is summarized in Table 10 and listed in Appendix A.3. Figure 47 shows test problem NS-3.

The fourth class of test problems are bound constrained problems with deterministic noise. The source of noise in noise corrupted design problems is due to the repeatable discretization and/or convergence error produced by a numerical simulation. For example, Newton's method might be used to obtain a solution to a nonlinear system of equations, and depending on the initial guess and the problem, there will be some error within a desired tolerance. An example is illustrated below. The square roots of all integers between one and one hundred (inclusive) are computed using Newton's method. The nonlinear system to be solved is

$$f(x) = x^2 - x_0 = 0 \tag{7.2}$$

where x_0 is the integer for which the square root is desired and x is the square root. The initial point is chosen as the approximation $1 + \frac{1}{2}(x_0 - 1)$, which is the Taylor expansion of the square root function to two terms about $x_0 = 1$. The tolerance value used to terminate Newton's method is $\epsilon = 0.01$, which is used as both the absolute and relative tolerance. The error between the Newton solution and the true square root (taken using MATLAB's `sqrt` function with double precision) is plotted in Figure 48. The figure shows that the absolute error decreases with increasing x_0 and the relative error remains somewhat constant. In theory, the relative error can grow with x_0, but this behavior will also depend on the solution method. In design problems, the relative error is typically used as the termination criterion, since it may not always be feasible to obtain a solution to low absolute error. It is also apparent that the error oscillates at a frequency which is a function of x_0. This type of behavior is exhibited in design problems that use iterative methods, as shown in Figure 41 (p. 96). Based on these observations, noise corrupted problems are defined as follows

[71]:

$$y_{\mathrm{n}}(x) = (1 + \epsilon_{\mathrm{n}}\phi(x))y(x) \tag{7.3}$$

where ϵ_{n} is the relative noise level and the noise function $\phi : \mathbb{R}^n \to [-1, 1]$ is defined in terms of the cubic Chebyshev polynomial $T_3(\cdot)$ by

$$\phi(x) = T_3(\phi_0(x)), \quad T_3(\alpha) = \alpha(4\alpha^2 - 3) \tag{7.4}$$

where

$$\phi_0(x) = 0.9 \sin(100\|x\|_1) \cos(100\|x\|_\infty) + 0.1 \cos(\|x\|_2) \tag{7.5}$$

The selected noise function combines high and low frequency oscillations and the resulting noise corrupted problem $y_{\mathrm{n}}(\cdot)$ exhibits noise that increases with increasing $|y(\cdot)|$, characteristic of the deterministic noise in design problems. Figure 49 shows $y_{\mathrm{n}}(x)$ with $\epsilon_{\mathrm{n}} = 0.001$ and $y(x) = 1 + \frac{1}{2}\|x - x_0\|_2^2$ with $x_0 = [0.5, 1]^T$. The noise corrupted problem set is summarized in Table 11 and listed in Appendix A.1.

Table 7: Test problem classification scheme of Hock and Schittkowski [49] corresponding to classification string (7.1)

O	Objective function
C	Constant
L	Linear
Q	Quadratic
S	Sum of squares
P	General polynomial
G	General nonlinear

C	Constraint functions (highest degree)
U	Unconstrained
B	Bounds
L	Linear
Q	Quadratic
P	General polynomial
G	General nonlinear

D	Regularity of the problem
R	Regular
I	Irregular

K	Information about the solution
T	Exact solution known (theoretical problem)
R	Exact solution unknown (design problem)

s	Serial number within class OCD-K
n	Number of variables
$m_{\mathcal{E}}$	Number of equality constraints
$m_{\mathcal{I}}$	Number of inequality constraints
b	Number of bound constraints

Table 8: Condensed list of bound constrained global optimization test problems from Appendix A.1.

	Problem name	Abbreviation	OCR	K	s	n	$m_{\mathcal{E}}$	$m_{\mathcal{I}}$	b
1	Schubert problem	SCHUBERT	GBR	T	1	1	0	0	2
2	Branin function	BRANIN	GBR	T	2	2	0	0	4
3	Camel back 3	CAMEL3	PBR	T	1	2	0	0	4
4	Camel back 6	CAMEL6	PBR	T	2	2	0	0	4
5	Dixon & Price function	DIXONPR	PBR	T	3	2	0	0	4
6	Goldstein & Price function	GOLDPR	PBR	T	4	2	0	0	4
7	Modified Langerman Problem	MLANGMN2	GBR	P	3	2	0	0	4
8	Modified Rosenbrock function	MROSEN	PBR	T	5	2	0	0	4
9	Paviani Problem	PAVIANI2	GBR	T	4	2	0	0	4
10	Shekel's foxholes	SHEKELF	GBR	P	5	2	0	0	4
11	Gulf R&D problem	GULFRD	SBR	T	1	3	0	0	6
12	Hartman 3	HARTMAN3	GBR	T	6	3	0	0	6
13	Shekel 5	SHEKEL5	GBR	T	7	4	0	0	8
14	Shekel 7	SHEKEL7	GBR	T	8	4	0	0	8
15	Shekel 10	SHEKEL10	GBR	T	9	4	0	0	8
16	Michalewicz Problem	MICH5	GBR	P	10	5	0	0	10
17	Paviani Problem	PAVIANI5	GBR	T	11	5	0	0	10
18	Hartman 6	HARTMAN6	GBR	T	12	6	0	0	12
19	Michalewicz Problem	MICH10	GBR	P	13	10	0	0	20
20	Paviani Problem	PAVIANI10	GBR	T	14	10	0	0	20

Table 9: Condensed list of nonlinearly constrained global optimization test problems from Appendix A.2.

	Problem name	Abbreviation	OCR	K	s	n	$m_{\mathcal{E}}$	$m_{\mathcal{I}}$	b
1	Constrained Schubert 1	CSCHUB1	GGR	T	1	1	0	1	2
2	Constrained Schubert 2	CSCHUB2	GGR	T	2	1	0	2	2
3	Hock & Schittkowski 9	hs009	GLR	T	1	2	1	0	4
4	Hock & Schittkowski 12	hs012	QQR	T	1	2	0	1	4
5	Hock & Schittkowski 14	hs014	QQR	T	2	2	1	1	4
6	Hock & Schittkowski 19	hs019	PQR	T	1	2	0	2	4
7	Multiple disconnected regions	mdc	LQR	T	1	2	0	1	4
8	Mystery function	MYST	GGR	T	3	2	0	1	4
9	Hock & Schittkowski 26	hs026	PPR	T	1	3	1	0	6
10	Hock & Schittkowski 32	hs032	QPR	T	1	3	1	1	6
11	Hock & Schittkowski 33	hs033	PQR	T	2	3	0	2	6
12	Phase and chemical equilibrium	CHEMEQ	GPR	P	1	3	2	0	6
13	Test problem 4	test4	LGR	T	1	3	1	3	6
14	Hock & Schittkowski 46	hs046	PGR	T	1	4	2	0	8
15	Hock & Schittkowski 81	hs081	GPR	P	2	5	3	0	10
16	Hock & Schittkowski 83	hs083	QQR	P	3	5	6	0	10
17	CSTR sequence design	CSTR	LGR	P	2	6	4	1	12
18	Hesse function	HESSE	QQR	T	4	6	0	6	12
19	Hock & Schittkowski 87	hs087	GGI	P	2	6	0	4	12
20	Hock & Schittkowski 100	hs100	PPR	P	2	7	0	4	14

Table 10: Condensed list of bound constrained nonsmooth global optimization test problems from Appendix A.3.

	Problem name	Abbreviation	OCR	K	s	n	$m_{\mathcal{E}}$	$m_{\mathcal{I}}$	b
1	One-dimensional problem	NS1D	PBR	T	1	1	0	0	2
2	Two-dimensional problem 1	NS2D1	GBR	T	1	2	0	0	4
3	Two-dimensional problem 2	NS2D2	GBR	T	2	2	0	0	4
4	Two-dimensional problem 3	NS2D3	GBR	T	3	2	0	0	4
5	Two-dimensional problem 4	NS2D4	GBR	T	4	2	0	0	4
6	Two-dimensional problem 5	NS2D5	GBR	T	5	2	0	0	4
7	Two-dimensional problem 6	NS2D6	GBR	T	6	2	0	0	4
8	Two-dimensional problem 7	NS2D7	GBR	T	7	2	0	0	4
9	Nonsmooth Hartman 3	NSHART3	GBR	T	8	3	0	0	6

Table 11: Condensed list of bound constrained noise corrupted global optimization test problems from Appendix A.1.

	Problem name	Abbreviation	OCR	K	s	n	$m_{\mathcal{E}}$	$m_{\mathcal{I}}$	b
1	Schubert problem	SCHUBERT	GBR	T	1	1	0	0	2
2	Branin function	BRANIN	GBR	T	2	2	0	0	4
3	Camel back 3	CAMEL3	PBR	T	1	2	0	0	4
4	Camel back 6	CAMEL6	PBR	T	2	2	0	0	4
5	Dixon & Price function	DIXONPR	PBR	T	3	2	0	0	4
6	Goldstein & Price function	GOLDPR	PBR	T	4	2	0	0	4
7	Paviani Problem	PAVIANI2	GBR	T	3	2	0	0	4
8	Shekel's foxholes	SHEKELF	GBR	P	4	2	0	0	4
9	Gulf R&D problem	GULFRD	SBR	T	1	3	0	0	6
10	Hartman 3	HARTMAN3	GBR	T	5	3	0	0	6

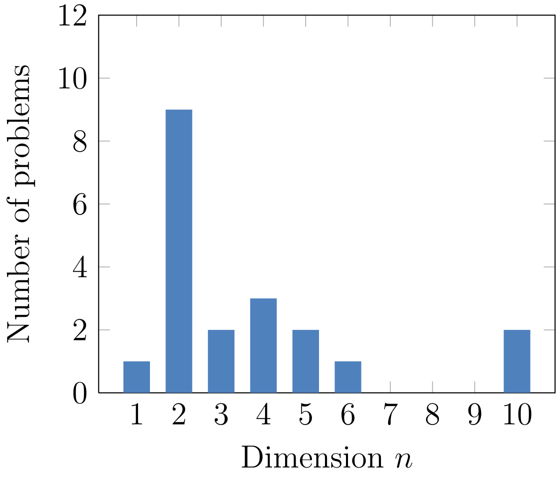


Figure 44: Number of bound constrained test problems of dimension n.

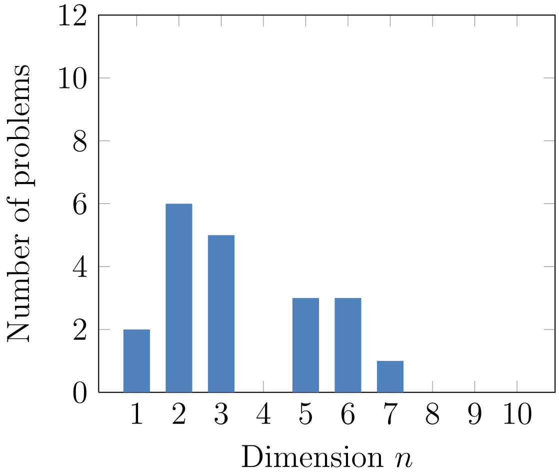


Figure 45: Number of nonlinearly constrained test problems of dimension n.

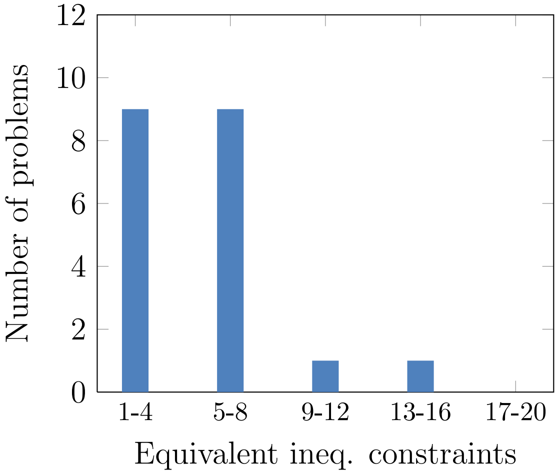


Figure 46: Size of nonlinearly constrained test problems in terms of equivalent number of inequality constraints.

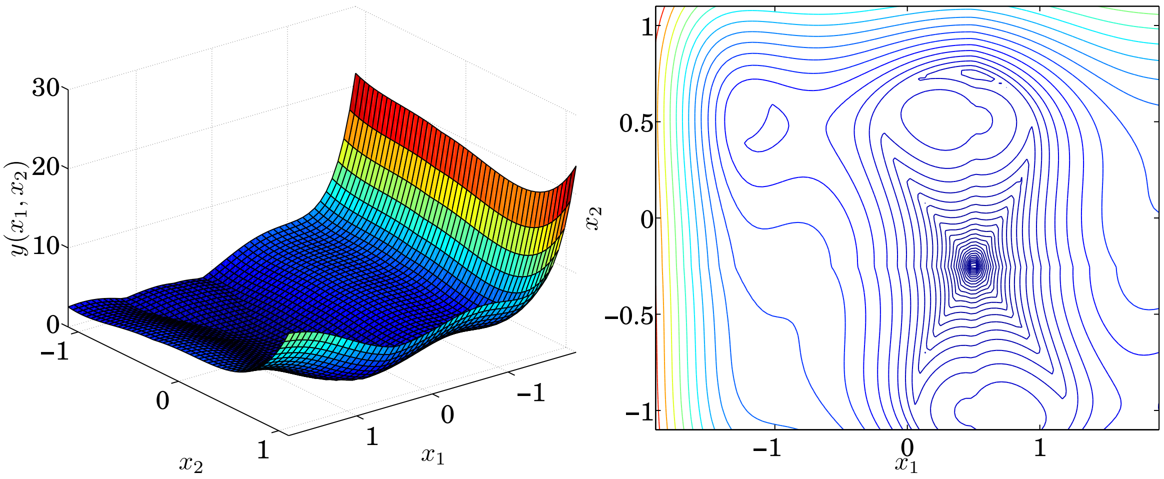


Figure 47: Test problem NS-3.

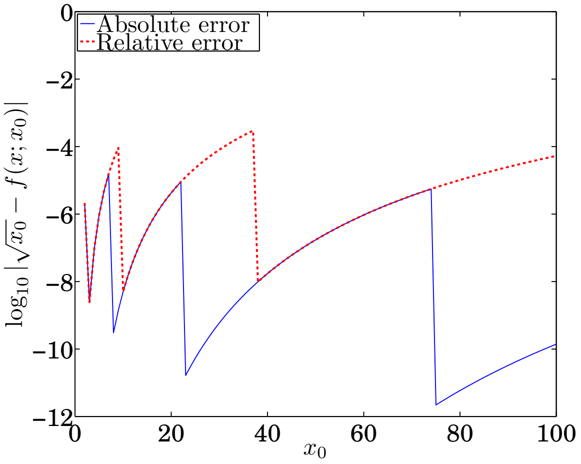


Figure 48: Error between MATLAB's `sqrt`(x_0) and the Newton approximation to $\sqrt{x_0}$; initial point chosen as the approximation $1 + \frac{1}{2}(x_0 - 1)$ and tolerance $\epsilon = 0.01$.

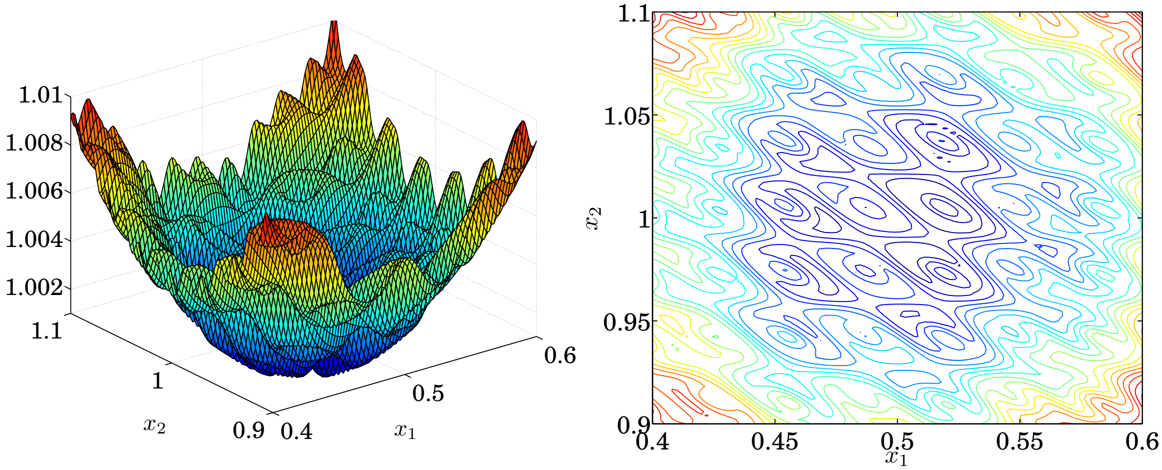


Figure 49: Noise corrupted quadratic function $y_n(x) = (1+0.001\phi(x))(1+\frac{1}{2}\|x-x_0\|_2^2)$ with $x_0 = [0.5, 1]^T$.

7.4 Competing Algorithms

Two main sets of competing algorithms will be considered: one set for bound constrained problems and one set for nonlinearly constrained problems. These algorithms are the state-of-the-art algorithms for surrogate model-based global optimization and were discussed in Chapter 3. The competing algorithms for the bound constrained problems are described in Table 12. Table 13 lists the competing algorithms for the nonlinearly constrained problems. Of the seven competing nonlinearly constrained algorithms, three employ the ℓ_1 smooth penalty function 4.1 to handle constraints: osEGO, RBF-G, and CORS-RBF. `DIRECT` has its own method for handling inequality constraints but equality constraints are included using the ℓ_1 smooth penalty function. The remaining algorithms handle constraints directly in their ISC.

The algorithms for the noise corrupted problems are listed in Table 14 and are based on the bound constrained algorithms. There is no straight-forward way to extend RBFs to the case of noise corrupted data [39], thus RBF-G is omitted and CORS-RBF must be substituted with CORS-GP, i.e., the CORS framework will employ a GP instead of a RBF for the case of noise corrupted problems.

The algorithms for the nonsmooth problems are listed in Table 15 and also based on the bound constrained algorithms with two options for the basis functions. The GP-based algorithms will employ the isotropic SE covariance function or the isotropic Matérn covariance function (equation (2.4)) with smoothness parameter $\nu = 1$. The RBF methods will employ the linear RBF in addition to the cubic RBF used thus far. The Matérn covariance function and the linear RBF are capable of modeling nonsmooth functions.

The ability of a surrogate model to capture nondifferentiable subspaces arises from the properties of the basis function. Since GPs and RBFs are linear combinations of basis functions, if the basis function has a derivative that is undefined at some point x, the surrogate model will be nondifferentiable at x. Consider the one dimensional case, i.e., $x \in \mathbb{R}$. The Matérn covariance function (2.3) with $\nu = 1$ is a function of $|x - x^\star|$. Thus, a GP model that employs this covariance function is continuous but not differentiable at $x^\star$. Similarly, the linear RBF (2.37) is a function of $\|x - x^\star\|_2$ which has an undefined first derivative at $x^\star$. The SE covariance function and the cubic RBF have derivatives which are defined everywhere and are considered "smooth" basis functions. Other options exist for the nonsmooth basis functions, e.g., the power exponential covariance function (2.2) with $p_h = 1, h = 1, \ldots, n$, but only one option for the smooth class and one option for the nonsmooth class are considered in this research.

Table 12: Competing algorithms for bound constrained test problems.

Algorithm	Basis function	Model fitting method	ISC
DIRECT	–	–	–
P-algorithm	Iso. SE GP	MLE	$\mathrm{P}(Y(x) < y^T)$
EGO	Iso. SE GP	MLE	$\mathbb{E}_I(x)$
osEGO	Iso. SE GP	–	$\log \ell(\theta, x^{(k+1)} \vert y^{(k)}, y^T)$
RBF-G	Cubic RBF	$Ax = b$	$h(\xi; y^T)$
CORS	Cubic RBF	$Ax = b$	Problem (3.43)
fBcEGO	Iso. SE GP	Proposed	$\mathbb{E}_\theta\left[\mathbb{E}_{I\vert\theta}(x)\right]$

Table 13: Competing algorithms for nonlinearly constrained test problems.

Algorithm	Basis function	Model fitting method	ISC
DIRECT	–	–	–
P	Iso. SE GP	MLE	$\mathrm{P}(Y(x) < y^T) \prod_{i=1}^{m} \mathrm{P}(C_i(x) \geq 0)$
EGO	Iso. SE GP	MLE	$\mathbb{E}_I(x) \prod_{i=1}^{m} \mathrm{P}(C_i(x) \geq 0)$
osEGO	Iso. SE GP	–	$\log \ell(\theta, x^{(k+1)} \vert y^{(k)}, y^T)$
RBF-G	Cubic RBF	$Ax = b$	$h(\xi; y^T)$
CORS	Cubic RBF	$Ax = b$	Problem (3.43)
fBcEGO	Iso. SE GP	Proposed	$\mathbb{E}_\theta \left[\mathbb{E}_{I_c \vert \theta}(x)\right]$

Table 14: Competing GP algorithms for noise corrupted test problems.

Algorithm	Basis function	Model fitting method	ISC
P-algorithm	Iso. SE GP	MLE	$\mathrm{P}(Y(x) < y^T)$
EGO	Iso. SE GP	MLE	$\mathbb{E}_I(x)$
osEGO	Iso. SE GP	MLE	$\log \ell(\theta, x^{(k+1)} \vert y^{(k)}, y^T)$
fBcEGO	Iso. SE GP	Proposed	$\mathbb{E}_\theta \left[\mathbb{E}_{I \vert \theta}(x)\right]$

Table 15: Competing algorithms for bound constrained nonsmooth test problems.

	Basis function			
Algorithm	Smooth	Nonsmooth	Model fitting method	ISC
P-algorithm	Iso. SE	Iso. Matérn, $\nu = 1$	MLE	$\mathrm{P}(Y(x) < y^T)$
EGO	Iso. SE	Iso. Matérn, $\nu = 1$	MLE	$\mathbb{E}_I(x)$
osEGO	Iso. SE	Iso. Matérn, $\nu = 1$	–	$\log \ell(\theta, x^{(k+1)} \vert y^{(k)}, y^T)$
RBF-G	Cubic RBF	Linear RBF	$Ax = b$	$h(\xi; y^T)$
CORS	Cubic RBF	Linear RBF	$Ax = b$	Problem (3.43)
fBcEGO	Iso. SE	Iso. Matérn, $\nu = 1$	Proposed	$\mathbb{E}_\theta \left[\mathbb{E}_{I \vert \theta}(x)\right]$

7.5 Metrics Of Performance

The material in this section is based on [71].

The goal of surrogate model-based global optimization algorithms is to find "good" solutions to expensive problems, i.e., to provide a significant reduction in the function value of a problem in as few function evaluations as possible. The performance of an algorithm can be defined in terms of a performance measure $t_{p,s} > 0$ obtained for each problem $p \in \mathcal{P}$ and each solver $s \in \mathcal{S}$ in their respective sets. Larger values of $t_{p,s}$ indicate worse performance. For example, this measure could be the analytical effort or the number of function evaluations to meet some termination criterion. Function evaluations are a natural metric of performance for benchmarking derivative-free solvers when the analytical work of the algorithm is negligible compared with the cost of one function evaluation. Users with expensive problems typically have some function evaluation budget to adhere to and are interested in the percentage of problems that s can solve to a given tolerance ϵ within τ function evaluations. This is the data profile and can be written as

$$d_s(\tau) = \frac{|\{p \in \mathcal{P} | t_{p,s} \leq \tau\}|}{|\mathcal{P}|} \tag{7.6}$$

As usual, there is some upper bound on the function evaluations allowed for testing, perhaps because a solver is showing signs of not converging, and $t_{p,s} = \infty$ if the convergence criterion is not satisfied within that limit.

The definition (7.6) is independent of the problem dimension n. This is not realistic because typically, the number of function evaluations needed to satisfy a convergence criterion grows superlinearly with n. Thus, the data profile of a solver $s \in \mathcal{S}$ is redefined as

$$d_s(\tau) = \frac{|\{p \in \mathcal{P} | \frac{t_{p,s}}{n+1} \leq \tau\}|}{|\mathcal{P}|} \tag{7.7}$$

The normalizing factor $n + 1$ enables interpretation of $d_s(\tau)$ as the percentage of problems that can be solved with the equivalent τ simplex gradients, $n + 1$ referring

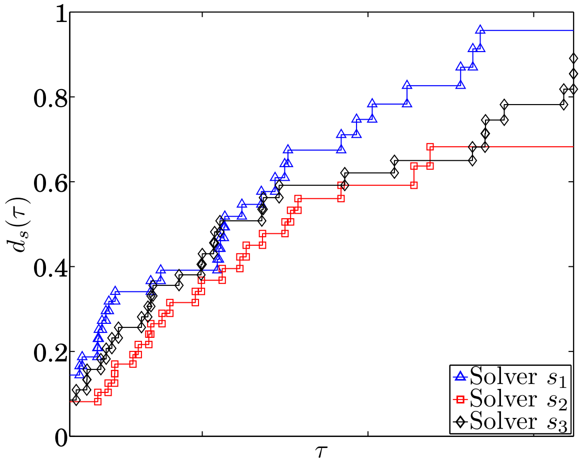


Figure 50: Notional data profile.

to the number of function evaluations needed to compute a one-sided finite difference estimate of the gradient. The scaling also prevents problems of low dimension from dominating the results. A notional data profile for three solvers is depicted in Figure 50. A solver s is said to dominate another solver $s^\star$ if the data profile of s lies completely above the data profile of $s^\star$, indicating that s solves a larger percentage of the test problems than $s^\star$ over all budgets considered. In Figure 50, s_1 dominates s_2.

The data profiler 7.6 with a fixed accuracy level ϵ answers the following questions:

1. If I have a budget of N simplex gradients, which solver is most likely to solve my problem to an accuracy of ϵ?

2. What percentage of problems can be solved to an accuracy of ϵ by a solver s, i.e., how robust is solver s for this accuracy level?

This type of data profiler also enables a user to study the influence of initial designs by solving the problems over a number of designs. A drawback of this type of profiler is that the tests are expensive, as a significant number of runs may not converge to the desired accuracy within the necessary function evaluation limit (to prevent the solver from running indefinitely). Furthermore, since a failed run is not considered, a user will not know what reduction in function value was obtained over the course of that run.

An alternative data profile involves fixing the number of simplex gradients allowed, i.e., specifying a budget, and reporting the percentage of problems for which s obtained an accuracy $\epsilon_{p,s}$ of at most τ. This is written as

$$d_s(r) = \frac{|\{p \in \mathcal{P} | \epsilon_{p,s} \leq \tau\}|}{|\mathcal{P}|} \tag{7.8}$$

The best value a solver can attain on a problem is $\epsilon_{p,s} = 0$, which means the global minimum was found exactly. The data profiler 7.8 with a fixed budget N answers the following questions:

1. If I have a budget of N simplex gradients, which solver is most likely to attain an accuracy of τ?

2. How reliable is a solver s for a given budget, i.e., what is the worst accuracy obtained over all problems?

This type of profiler enables a user with a budget to determine what solver is most likely to attain a τ-global minimizer. Unlike the first profiler, this second profiler includes all runs in the results, except for nonlinearly constrained problems, which may fail by not finding a feasible value by the time the budget is exhausted. A drawback of this profiler is that it will not return the success rate of a solver. In addition, small initial designs must be used with this profiler so that the initial design does not exhaust the fixed budget and so the accuracy attained is a result of the solver and not of a systematically-placed sample.

It is clear that both types of profilers described above report different information. In some cases, one profiler will be preferred over the other, or both may be used to evaluate a hypothesis. Naturally, as with any experiment, there may be outliers in the results. The median and spread of the performance of an algorithm is also of interest. For this purpose, box plots will be used to quantify this information and to identify outliers.

7.6 Experiments & Test Matrices

The experiments which are designed to evaluate the hypotheses in §7.1 are outlined in this section. The experiments make reference to the information provided thus far in the chapter. Test matrices for each experiment are given in Figures 51 – 53 and Table 16.

Experiment 1

Experiment 1.1: Record the simplex gradients $N_{p,s}$ required to solve the bound constrained problems (Table 8) using the smooth basis functions listed in Table 15 to within 1% of the global minimum over the nine initial designs in Table 5. Enforce an upper limit of $N = 50$, at which point an algorithm is considered to have failed to solve a problem.

Experiment 1.2: Record the simplex gradients $N_{p,s}$ required to solve the nonsmooth problems (Table 10) using the nonsmooth basis functions in Table 15 to within 1% of the global minimum over the nine initial designs in Table 5. Enforce an upper limit of $N = 50$, at which point an algorithm is considered to have failed to solve a problem.

Experiment 1.3: Record the simplex gradients $N_{p,s}$ required to solve the bound constrained problems (Table 8) using the nonsmooth basis functions listed in Table 15 to within 1% of the global minimum over the nine initial designs in Table 5. Enforce an upper limit of $N = 50$, at which point an algorithm is considered to have failed to solve a problem.

Experiment 2

Experiment 2.1: Record the accuracy $\epsilon_{p,s}$ attained by the bound constrained algorithms after solving the bound constrained test problems (Table 8) with $N = 5$, 10, and 20 simplex gradients starting with FCP initial design.

Experiment 2.2: Record the simplex gradients $N_{p,s}$ required to solve bound constrained problems to within 1% and 0.1% of the global minimum over the nine initial designs in Table 5. Enforce an upper limit of $N = 50$, at which point an algorithm is considered to have failed to solve a problem.

Experiment 3

Experiment 3.1: Record the accuracy $\epsilon_{p,s}$ attained by the nonlinearly constrained algorithms after solving the nonlinearly constrained test problems (Table 9) with $N = 5$, 10, and 20 simplex gradients starting with FCP initial design.

Experiment 3.2: Record the equality constraint violation $\sum_{i \in \mathcal{E}} |c_i(x)|$ attained by the nonlinearly constrained algorithms after solving the nonlinearly constrained test problems (Table 9) with $N = 5$, 10, and 20 simplex gradients starting with FCP initial design.

Experiment 4: Record the accuracy $\epsilon_{p,s}$ attained by the noise corrupted algorithms (Table 14) after solving the noise corrupted test problems (Table 11) with $N = 5$, 10, and 20 simplex gradients and with noise levels $\epsilon_{\mathrm{n}} = 0\%$, 0.01%, and 1% starting with the FCP initial design.

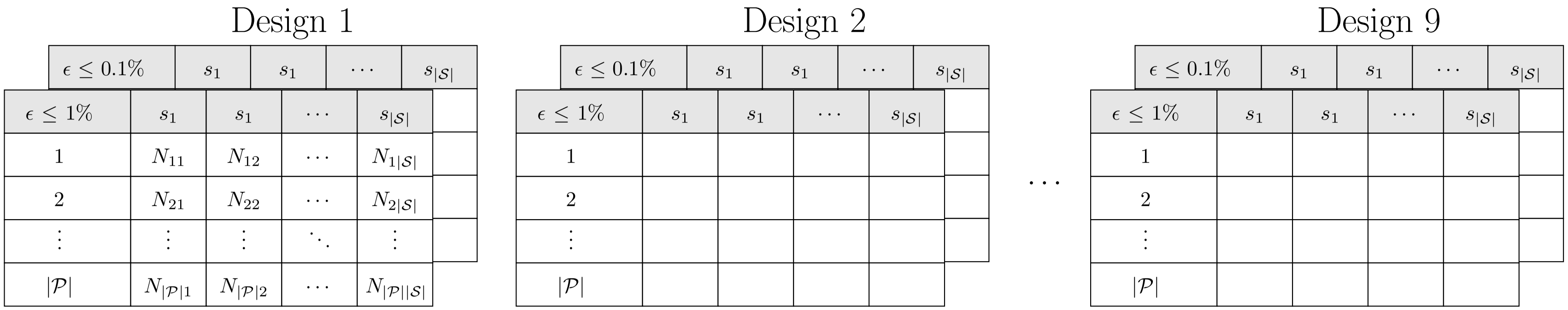
Experiment 5

Experiment 5.1: Same as Experiment 1.2.

Experiment 5.2: Same as Experiment 1.3.

Experiment 5.3: Record the simplex gradients $N_{p,s}$ required to solve the nonsmooth problems (Table 8) using the smooth basis functions listed in Table 15 to within 1% of the global minimum over the nine initial designs in Table 5. Enforce an upper limit of $N = 50$, at which point an algorithm is considered to have failed to solve a problem.

Experiment 6: Test the performance of `fBcEGO` on a practical problem with hard constraints using predictor imputation, penalized imputation, and maximum value imputation for two different initial designs selected according to the results of Experiment 1. For each method, record the accuracy ϵ attained after $N = 50$ simplex gradients and the percentage of iterates beyond the initial design that failed.



Design 1

$\epsilon \leq 0.1\%$	s_1	s_1	$\cdots$	$s_{\lvert\mathcal{S}\rvert}$

$\epsilon \leq 1\%$	s_1	s_1	$\cdots$	$s_{\lvert\mathcal{S}\rvert}$
1	N_{11}	N_{12}	$\cdots$	$N_{1\lvert\mathcal{S}\rvert}$
2	N_{21}	N_{22}	$\cdots$	$N_{2\lvert\mathcal{S}\rvert}$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
$\lvert\mathcal{P}\rvert$	$N_{\lvert\mathcal{P}\rvert 1}$	$N_{\lvert\mathcal{P}\rvert 2}$	$\cdots$	$N_{\lvert\mathcal{P}\rvert\lvert\mathcal{S}\rvert}$

Design 2

$\epsilon \leq 0.1\%$	s_1	s_1	$\cdots$	$s_{\lvert\mathcal{S}\rvert}$

$\epsilon \leq 1\%$	s_1	s_1	$\cdots$	$s_{\lvert\mathcal{S}\rvert}$
1				
2				
$\vdots$				
$\lvert\mathcal{P}\rvert$				

$\cdots$

Design 9

$\epsilon \leq 0.1\%$	s_1	s_1	$\cdots$	$s_{\lvert\mathcal{S}\rvert}$

$\epsilon \leq 1\%$	s_1	s_1	$\cdots$	$s_{\lvert\mathcal{S}\rvert}$
1				
2				
$\vdots$				
$\lvert\mathcal{P}\rvert$				

Figure 51: Test matrices for Experiment 1 (front row only), Experiment 2.2, and Experiment 5 (front row only).

$N = 5$

	s_1	s_2	$\cdots$	$s_{\lvert\mathcal{S}\rvert}$
1	ϵ_{11}	ϵ_{12}	$\cdots$	$\epsilon_{1\lvert\mathcal{S}\rvert}$
2	ϵ_{21}	ϵ_{22}	$\cdots$	$\epsilon_{2\lvert\mathcal{S}\rvert}$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
$\lvert\mathcal{P}\rvert$	$\epsilon_{\lvert\mathcal{P}\rvert 1}$	$\epsilon_{\lvert\mathcal{P}\rvert 2}$	$\cdots$	$\epsilon_{\lvert\mathcal{P}\rvert\lvert\mathcal{S}\rvert}$

$N = 10$

	s_1	s_2	$\cdots$	$s_{\lvert\mathcal{S}\rvert}$
1				
2				
$\vdots$				
$\lvert\mathcal{P}\rvert$				

$N = 20$

	s_1	s_2	$\cdots$	$s_{\lvert\mathcal{S}\rvert}$
1				
2				
$\vdots$				
$\lvert\mathcal{P}\rvert$				

Figure 52: Test matrices for Experiment 2.1 and Experiment 3; for Experiment 3 the equality constraint violation is recorded in addition to the accuracy attained.

$N = 5$

$\epsilon_n = 1\%$	s_1	s_2	$\cdots$	$s_{\lvert\mathcal{S}\rvert}$

$\epsilon_n = 0.01\%$	s_1	s_2	$\cdots$	$s_{\lvert\mathcal{S}\rvert}$

$\epsilon_n = 0\%$	s_1	s_2	$\cdots$	$s_{\lvert\mathcal{S}\rvert}$
1	ϵ_{11}	ϵ_{12}	$\cdots$	$\epsilon_{1\lvert\mathcal{S}\rvert}$
2	ϵ_{21}	ϵ_{22}	$\cdots$	$\epsilon_{2\lvert\mathcal{S}\rvert}$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
$\lvert\mathcal{P}\rvert$	$\epsilon_{\lvert\mathcal{P}\rvert 1}$	$\epsilon_{\lvert\mathcal{P}\rvert 1}$	$\cdots$	$\epsilon_{\lvert\mathcal{P}\rvert\lvert\mathcal{S}\rvert}$

$N = 10$

$\epsilon_n = 1\%$	s_1	s_2	$\cdots$	$s_{\lvert\mathcal{S}\rvert}$

$\epsilon_n = 0.01\%$	s_1	s_2	$\cdots$	$s_{\lvert\mathcal{S}\rvert}$

$\epsilon_n = 0\%$	s_1	s_2	$\cdots$	$s_{\lvert\mathcal{S}\rvert}$
1				
2				
$\vdots$				
$\lvert\mathcal{P}\rvert$				

$N = 20$

$\epsilon_n = 1\%$	s_1	s_2	$\cdots$	$s_{\lvert\mathcal{S}\rvert}$

$\epsilon_n = 0.01\%$	s_1	s_2	$\cdots$	$s_{\lvert\mathcal{S}\rvert}$

$\epsilon_n = 0\%$	s_1	s_2	$\cdots$	$s_{\lvert\mathcal{S}\rvert}$
1				
2				
$\vdots$				
$\lvert\mathcal{P}\rvert$				

Figure 53: Test matrices for Experiment 4.

Table 16: Test matrix for Experiment 6.

Imputation method	Initial design	Initial failures	Subsequent failures	$y_{\min}$
Predictor	Design 1			
	Design 2			
Penalized	Design 1			
	Design 2			
Max. value	Design 1			
	Design 2			

CHAPTER VIII

RESULTS

8.1 Experimental Setup & Implementation Details

8.1.1 Experimental Designs

The experimental designs used in this research were outlined in §5.1. Specifically, experiments which are designed to determine the influence of the initial design solve problems over the nine designs in Table 5. Experiments which determine the accuracy or equality constraint violation attained on a fixed budget use the FCP design. With the exception of the problem specific design DGS and its combinations, the designs are standard and only vary with dimension, i.e., they are not randomized designs such as those generated by MATLAB's `lhsdesign`. Thus, the designs themselves will not generate any variability in the results. Points in the initial design which satisfy

$$\|x^{(i)} - x^*\|_2 \leq 0.01 \min_{1\leq h\leq n} (x_h^u - x_h^\ell), \qquad i = 1, \ldots, k \tag{8.1}$$

are removed from the initial design. That is, points in the initial design $\mathcal{D}^{(1)}$ which fall within a ball of radius equal to 1% of the smallest dimension of the *unnormalized* space $A = \{x|x^\ell \leq x \leq x^u\}$ of *all* the global minimizers x^* are excluded from $\mathcal{D}^{(1)}$. This is to prevent algorithms from "stumbling upon" the solution.

8.1.2 Model Fitting

For algorithms based on GP models and requiring hyperparameter estimation (P-Algorithm and EGO), the MLE approach was used within a gradient-based multi-start framework of $10I + 1$ maximin Latin hypercube points plus 2^I corners, where I is the number of hyperparameters; for noise-free problems, $I = 2$ and for noise corrupted problems $I = 3$. The model fitting problem (problem (2.15)) was reformulated such that the hyperparameters θ were searched over the log-space, eliminating the nonnegativity constraints $\theta > 0$. The space A defined by the bound constraints was normalized to the unit hypercube to facilitate the intelligent selection of θ values for the multistart solution. The initial values for θ satisfied $10^{-3} \leq \theta_0, \theta_1 \leq 2$ and $10^{-4} \leq \theta_n \leq 0.2$. The gradients were computed using equation (2.18). MATLAB's `fminunc` was used as the primary solver.

RBF-G and CORS-RBF used cubic RBF models, which were fit by solving a linear system of equations, with safeguards to improve conditioning and handle singularities.

8.1.3 Optimization Of Infill Sampling Criteria

For the optimization of the ISC, it was found that a gradient-based multistart strategy was prohibitively expensive for large n. Thus, `DIRECT` was used with $100n$ function evaluations to maximize the ISC of each algorithm, and a gradient-based solver was used to refine the best solution found. CORS-RBF used a pre-generated $10,000$ point LHD to sample the ISC and a gradient-based local solver was used to refine the best solution.

For the improvement-based algorithms, ISC gradients were computed using central finite differences; for RBF-G, analytical derivatives for $h_k(\cdot; y^T)$ were derived using the relations in §3.3.2. For CORS-RBF, analytical gradients of the maximin constraints and the cubic RBF were also provided.

For algorithms that required a surface minimum (P-algorithm, osEGO, and RBF-G), values of $s_{\min}$ were set to the minimum sample value $y_{\min}$ if there were no nonlinear constraints. If nonlinear constraints were present and no feasible samples were found, $s_{\min}$ was chosen to minimize the ℓ_1 inequality constraint violation. The subsolvers used to maximize the ISC for each algorithm were consistent and deterministic within each test set and thus contributed no variability to the results.

8.1.4 Target Values

The P-algorithm, osEGO, and RBF-G require a target value y^T which controls the global-local balance. The approach from [44] is used. This approach is outlined in Algorithm 2. A single target value is used at each iteration and the value is cycled such that the search starts off fairly global and after a few iterations becomes fairly local. This cycle repeats until convergence. CORS requires a cycle for the maximin radius and this cycle was taken as [97]

$$\langle 0.95, 0.25, 0.05, 0.03, 0 \rangle \tag{8.2}$$

8.1.5 Stopping Criteria

All the experiments in §7.6 either use the accuracy attained as the stopping criterion or record the accuracy attained after a simplex gradient budget is exhausted. For bound constrained, noise corrupted, and nonsmooth problems, this stopping criterion only depends on $y_{\min}$ and y^*:

$$\epsilon = \begin{cases} \left| y_{\min} - y^* \right| / \left| y^* \right|, & |y^*| > 0 \\ y_{\min}, & y^* = 0 \end{cases} \tag{8.3}$$

Because a feasible value for $y_{\min} = \min_{1 \le i \le k} y(x^{(i)})$ is always available for problems without nonlinear constraints, this value can be updated at every iteration and thus experiments which record the accuracy attained after a budget is exhausted will return no failures. For nonlinearly constrained problems, $y_{\min}$ is updated by the rule

$$y_{\min} = \min_i y(x^{(i)}), \qquad \{i : c_j(x^{(i)}) \ge 0, 1 \le i \le k, 1 \le j \le m\} \tag{8.4}$$

There may not be a feasible value of $y_{\min}$ available at each iteration, thus experiments that record the accuracy attained after a budget is exhausted may return some failures. The equality constraint violation $\sum_{i \in \mathcal{E}} |c_i(x)|$ is treated as a secondary metric for nonlinearly constrained problems, since any relaxation factor as in (1.2) is arbitrary.

8.1.6 Design Space Symmetry

It was found that some algorithms could solve the low-dimensional symmetric problems in one iteration. This is because the FCP initial design is symmetric, which led to a symmetric surrogate model and hence a symmetric ISC with a maximum at the center of the design space. To provide a fair comparison for all algorithms, any symmetries in the design space were removed by applying a linear shift to x. Test problem NS-3 (Figure 47), for example, required a linear shift to remove the symmetry.

8.2 Experiment 1: Influence Of Initial Design On Performance

Experiments were conducted to determine the effect of the initial design on the performance of each algorithm. Tests were performed on smooth bound constrained problems using smooth basis functions (Experiment 1.1), on nonsmooth problems using nonsmooth basis functions (Experiment 1.2), and on smooth bound constrained problems using nonsmooth basis functions (Experiment 1.3). A total of 348 tests were performed for each of the nine designs listed in Table 5. All problems were $n \leq 3$. There were no significant differences in the number of successful tests for each design; these data are presented in Table 17. The results are compiled and presented as a set of data profiles and box plots in Figure 54. Outliers (represented by crosses) are taken as data points that fall outside of the range [99]

$$[Q_1 - c(Q_3 - Q_1), Q_2 - c(Q_3 - Q_1)] \tag{8.5}$$

with Q_i being the ith quartile and $c = 1.5$.

In terms of performance on the data profile, design 2 (N_1 DGS) and design 7 (N_2 DGS+FCP) dominate the remaining designs but for different budgets. Thus if a user has a budget of $N \leq 22$, design 2 (N_1 DGS) is more likely to solve a problem to within $\epsilon = 1\%$ accuracy, whereas if a user has a budget of $22 \leq N \leq 50$, design 7 (N_2 DGS+FCP) is preferred, followed closely by design 5 (N_2 LHD). The upper limit of $N = 50$ was enforced to prevent algorithms from running indefinitely. The superior performance of design 2 and design 7 on the data profiler is due to the fact that both designs utilized the deterministic global solver (DGS) `DIRECT` to place initial points intelligently rather than systematically using a static design. `DIRECT` only places the first three samples systematically, and then uses this and all previous information to place new samples sequentially in promising areas. This result indicates the importance of placing samples intelligently as soon as possible.

In terms of median number of simplex gradients required to solve a problem to within $\epsilon = 1\%$ accuracy, design 2 (N_1 DGS) and design 6 (N_1 DGS+FCP) exhibit the best performance. These two designs solved all problems to an accuracy of 1% in a median value of approximately ten simplex gradients and also exhibit a moderate inter-quartile range. The performance can be explained in the same manner as in the previous paragraph.

Design 3 (N_2 DGS) and design 7 (N_2 DGS+FCP) also incorporate a deterministic global solver, but their median performance is significantly worse. The difference between each pair of designs is the number of initial points. The reader is reminded that $N_1 = (n+1)(n+2)/2$ and $N_2 = 10n+1$. For $n \leq 3$, $N_1 < N_2$ by a large margin. Thus, the designs with the *lower* number of initial points performed better, because the more sophisticated surrogate model-based algorithms can take over sooner. For $n > 17$, $N_2 < N_1$ and it is inferred that N_2 number of initial samples should be used. Note the importance of having both the data profiles and the box plots: while design 7 (N_2 DGS+FCP) had the worst median performance when taken over all problems, it dominated the competing designs for $22 \leq N \leq 50$.

The next part of the analysis involves FCP designs. Designs 6, 7, 8, and 9 are identical to designs 2, 3, 4, and 5, respectively, but augmented with an FCP design. It is seen from Figure 54 that there were no significant differences in the location

Table 17: Successful tests as a function of initial design

Design	1	2	3	4	5	6	7	8	9
Successful tests (out of 348)	268	279	289	269	293	283	303	274	291

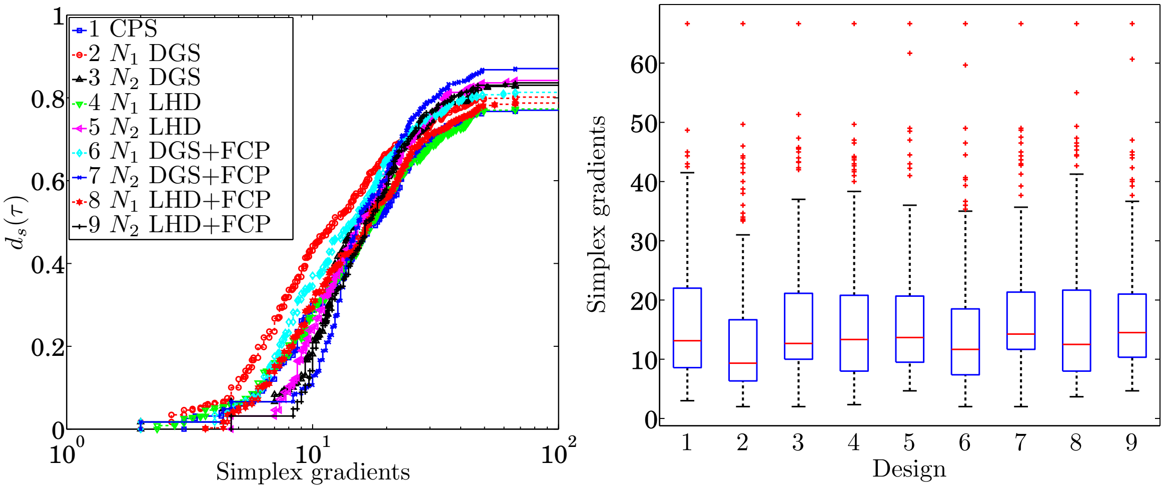


Figure 54: Effect of initial design on performance of algorithms; problems with $n \leq 3$ solved to 1% accuracy.

or variation of the performance when a design was augmented with the FCP design. It was demonstrated in §3.1 that some surrogate model-based algorithms will often sample the corner points and along the boundary of the design space first because the uncertainty is greatest in these areas. Thus, including the corner points may save some analytical time. For problems with hard constraints and/or unknown bounds, it is recommended that the corner points be included in the initial design. The hard constraint handling techniques discussed in §5.3 will penalize the corner points if they return failed iterations, driving the search away from these regions.

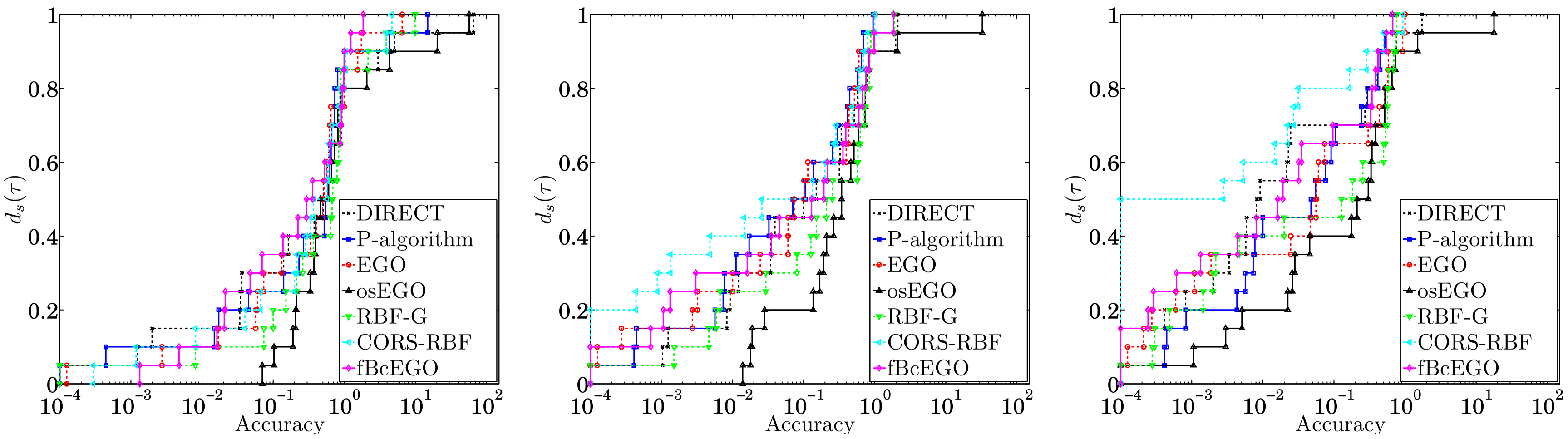
With respect to the specifics of this experiment, it was hypothesized in §7.1 that design 6 would have the best median performance in terms of simplex gradients. Given what was learned about FCP-augmented designs for surrogate model-based global optimization, the results support the hypothesis.

8.3 Experiment 2: Performance Of ISC On Bound Constrained Problems

This section presents the results of the competing bound constrained algorithms (Table 12) on the bound constrained test problems (Table 8). Tests were performed with a function evaluation budget of $N(n+1)$ function evaluations, with $N = 5$, 10, and 20, and the accuracy $\epsilon_{p,s}$ attained was recorded. The results are shown in Figures 55 and 56.

For $N = 5$, `fBcEGO` demonstrates superior performance for $0.02 \leq \epsilon \leq 0.5$. The remaining algorithms with the exception of osEGO exhibit similar performance with each other. For this case, `fBcEGO` is the most robust, i.e., solves all problems to within the lowest ϵ, excluding outliers (note that for this case, `fBcEGO` had no outliers, as shown in Figure 65a. osEGO performs poorly because this algorithm generally requires a large number of function evaluations. Difficulties in maximizing osEGO's ISC have also degraded the true performance. For $N = 10$, the trends are similar but with CORS-RBF dominating most the competing algorithms for $0 \leq \epsilon \leq 0.02$. In this case, CORS-RBF, EGO, P-algorithm, and `fBcEGO` are the most robust. `fBcEGO` had one outlier, as shown in Figure 65b, which can but excluded from the data profile by ignoring the uppermost step in Figure 64b. For $N = 20$, CORS-RBF completely dominates the competing algorithms with the exception of two outliers. CORS-RBF can achieve superior accuracy, solving 50% of the problems to $\epsilon \leq 10^{-4}$. `fBcEGO` demonstrates above average performance relative to the remaining algorithms as well as superior robustness, tying with the P-algorithm.

In terms of median accuracy attained, `fBcEGO` exhibits relatively good performance with the best median accuracy for $N = 5$ and competitive accuracy for $N = 10$ and $N = 20$. For these two latter cases, CORS-RBF exhibited the best median accuracy. For this experiment, `fBcEGO` exhibited competitive performance but as N became larger, CORS-RBF dominated the competing algorithms. The results do not support the hypothesis in §7.1.

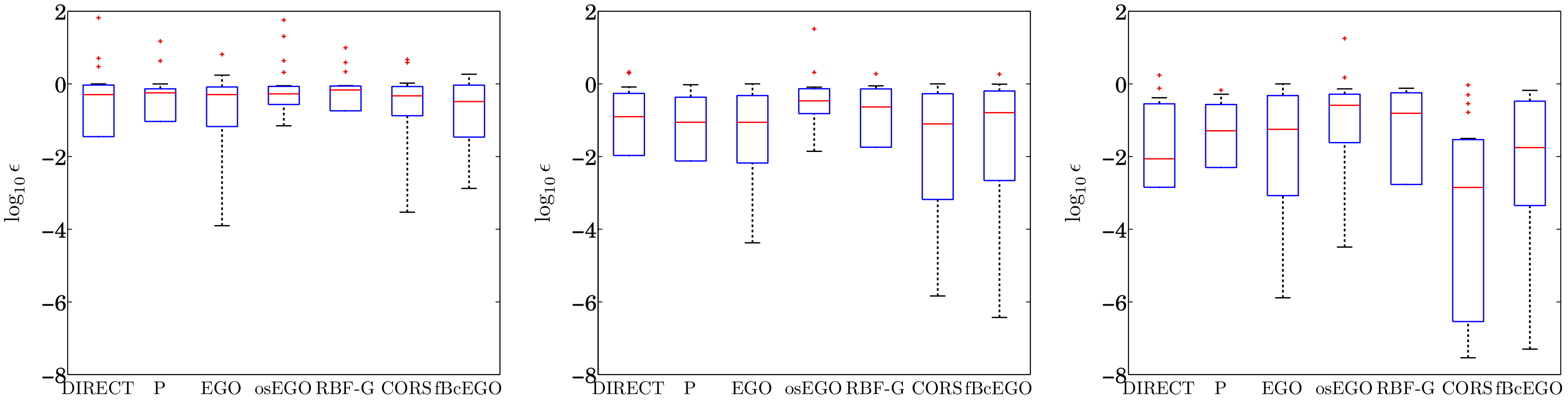


(a) $N = 5$

(b) $N = 10$

(c) $N = 20$

Figure 55: Results of bound constrained tests; algorithms were executed for a maximum of $N = 5$, 10, and 20 simplex gradients and the accuracy attained was recorded.



(a) $N = 5$

(b) $N = 10$

(c) $N = 20$

Figure 56: Box plots of bound constrained tests with accuracy metric as a function of budgeted simplex gradients N.

The ability of the bound constrained algorithms to solve problems to high degrees of accuracy was tested next. The algorithms were tested on ten bound constrained problems with $n \leq 3$ (Table 18) over nine initial designs (Table 5), giving a total of 90 tests per algorithm. Two sets of these tests were performed: one set for $\epsilon = 1\%$ and one set for $\epsilon = 0.1\%$. An upper limit of $N = 50$ is enforced, at which point a test is considered a failure. The selected problems were solvable by all algorithms and the low dimensionality allowed the tests to complete within a reasonable time frame. As a benchmark, the tests completed in approximately 72 hours on a 64-bit Windows 7 PC with 4GB RAM and a quad-core Intel i5 750 processor running at 2.67 GHz. One instance of MATLAB 7.9 was running with multithreading enabled.

The results are shown in Figures 57 and 58. For the first set of tests with $\epsilon \leq 1\%$, EGO, `fBcEGO`, and CORS-RBF exhibited similar performance which was also superior to the remaining algorithms. In this case, EGO and CORS-RBF were the most reliable algorithms, solving the largest percentage of problems within the upper limit of $N = 50$. EGO, `fBcEGO`, and CORS-RBF show similar location and variation for the performance (see Figure 58). For tests with $\epsilon \leq 0.1\%$, CORS-RBF dominated all competing profiles and proved to be the most reliable, solving nearly 80% of all problems. EGO and `fBcEGO` followed CORS-RBF closely up to $N = 10$ but showed inferior performance for higher N. All of the remaining algorithms with the exception of osEGO showed similar reliability. Considering only the successful cases, EGO, `fBcEGO`, and CORS-RBF again show similar location and variation for the performance (see Figure 58).

Contrary to the hypothesis in §7.1, `fBcEGO` did not provide any significant performance gain over the MLE-based EGO, although both `fBcEGO` and EGO performed significantly better than the P-algorithm. The similar performance between EGO and `fBcEGO` is explained in two ways. First, the bound constrained test problems were smooth and generally well-behaved; the MLE approach works well for smooth, well-behaved problems. Second, the implementations of the competing algorithms in the current research are highly optimized and robust. Asymptotic expansions of the ISCs were used for the P-algorithm and EGO, while analytical gradients were used for the ISCs of RBF-G and CORS-RBF. Neither asymptotic expansions nor analytical gradients could be derived for the fully Bayesian case.

A problem specific analysis is reported in Table 19 and Table 20 for $\epsilon = 0.01$ and $\epsilon = 0.001$, respectively. In general, the most difficult problems were the Goldstein & Price function, the Paviani problem, and Shekel's Foxholes. The Goldstein & Price function has large discrepancies in function values but this can be corrected with a logarithmic transformation; it is the shape of the global minimum basin that is problematic. The Paviani problem has very small discrepancies in function values, which may cause an algorithm to become trapped in flat regions that are not critical points. Shekel's Foxholes is a "needle in the haystack" type of problem with deep basins that have relatively small diameters. The Gulf R&D problem is not difficult but it shows significant failure rates. For this problem, $y^* = 0$ and it is generally more difficult to meet an absolute tolerance than a relative tolerance.

Table 18: Low dimensional design problems used for tests over different initial designs.

	Test Function	Abbreviation	n	A	No. of local/global min.	$y(x^*)$
1	Schubert	SCHUBERT	1	$[-10, 10]$	19/3	-12.0312494421671
2	Branin	BRANIN	2	$[-5, 10] \times [0, 15]$	3/3	0.397887357729738
3	Camel back 3	CAMEL3	2	$[-1.9, 1.9] \times [-1.1, 1.1]$	3/1	0.
4	Camel back 6	CAMEL6	2	$[-1.9, 1.9] \times [-1.1, 1.1]$	6/2	-1.03162845348987
5	Dixon-Price	DIXONPR	2	$[-1, 2] \times [-1, 1]$	1/1	0.
6	Goldstein & Price[a]	GOLDPR	2	$[-2, 2]^2$	4/1	3.
7	Paviani	PAVIANI2	2	$[2.001, 9.999]^2$	4/1	4.98150982324376
8	Shekel's foxholes[b]	SHEKELF	2	$[0, 10]^2$	15/1	-12.1190083797535
9	Gulf R&D	GULFRD	3	$[0.1, 100] \times [0, 25.6] \times [0, 5]$	unknown/1	0.
10	Hartman 3	HARTMAN3	3	$[0, 1]^3$	4/1	-3.862782147820756

[a] log-transformed function $\log y$ was used [56]

[b] log-transformed function $-\log(-y)$ was used

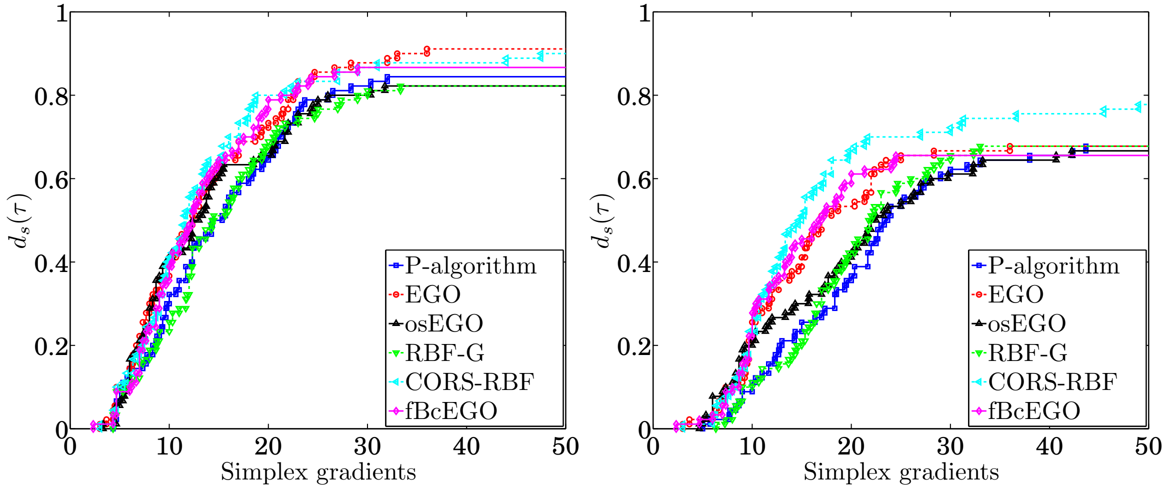


Figure 57: Performance of competing algorithms on ten bound constrained problems with $n \leq 3$ over nine initial designs (Table 5); percentage of problems solved to $\epsilon \leq 1\%$ (left) and $\epsilon \leq 0.1\%$ (right) as a function of simplex gradients.

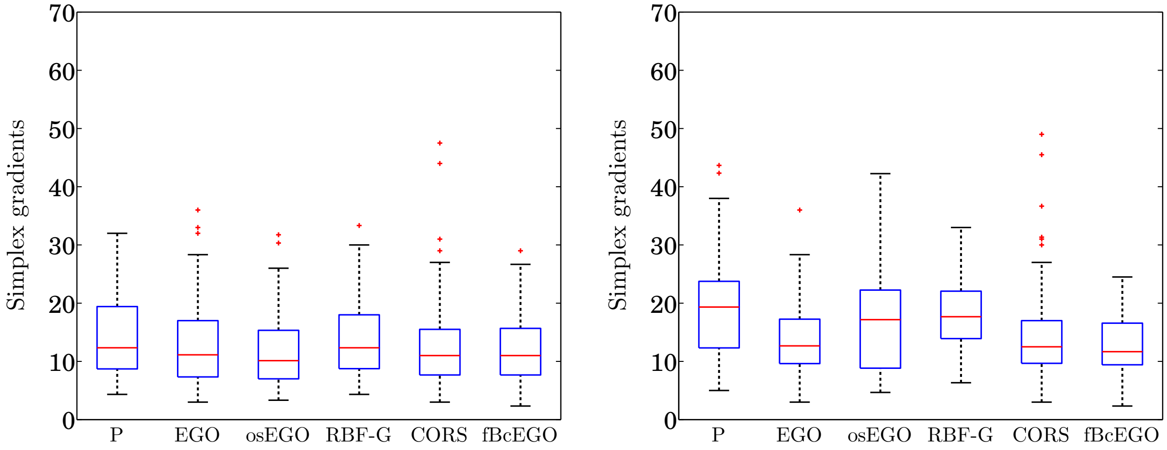


Figure 58: Box plots of test results of ten bound constrained problems with $n \leq 3$ over nine initial designs (Table 5); $\epsilon \leq 1\%$ (left) and $\epsilon \leq 0.1\%$ (right). Successful cases only.

Table 19: Number of function evaluations to achieve an accuracy of 1%; low dimensional problems.

	P-algorithm				**EGO**				**osEGO**				**RBF-G**				**CORS-RBF**				**fBcEGO**			
Problem	Fail %	mean	min	max	Fail %	mean	min	max	Fail %	mean	min	max	Fail %	mean	min	max	Fail %	mean	min	max	Fail %	mean	min	max
SCHUBERT	0.00	41.78	34	53	0.00	35.56	22	46	0.00	30.89	12	52	0.00	37.89	32	54	33.33	61.17	36	95	0.00	34.33	22	46
BRANIN	0.00	47.67	28	85	11.11	38.38	28	50	0.00	51.00	29	75	0.00	52.78	36	75	0.00	35.89	25	55	0.00	36.67	29	45
CAMEL3	0.00	19.67	13	29	0.00	17.22	9	30	0.00	19.78	10	27	0.00	18.89	13	31	0.00	16.67	9	27	0.00	18.22	7	30
CAMEL6	0.00	28.89	15	41	0.00	25.78	17	38	0.00	26.44	14	41	0.00	28.33	17	41	0.00	26.22	17	38	0.00	28.00	20	40
DIXONPR	0.00	28.00	14	48	0.00	23.89	14	44	0.00	23.89	18	34	0.00	27.78	14	37	0.00	24.33	14	40	0.00	21.44	14	30
GOLDPR	22.22	49.29	33	70	33.33	57.83	22	96	77.78	33.00	24	42	44.44	41.00	33	49	0.00	36.33	22	56	33.33	37.83	26	58
PAVIANI2	33.33	59.50	14	96	11.11	59.63	14	108	33.33	48.17	14	91	11.11	46.25	14	100	11.11	45.25	14	81	33.33	43.00	14	80
SHEKELF	66.67	55.33	37	71	0.00	48.56	18	85	66.67	42.33	26	62	33.33	75.50	58	90	33.33	45.00	27	93	22.22	59.29	30	87
GULFDR	33.33	77.67	57	91	33.33	71.83	56	88	0.00	71.78	48	127	88.89	58.00	58	58	22.22	62.14	39	71	44.44	69.20	50	85
HARTMAN3	0.00	40.11	23	57	0.00	36.33	18	55	0.00	41.89	22	62	0.00	49.89	28	68	0.00	47.89	29	75	0.00	41.67	25	59

Table 20: Number of function evaluations to achieve an accuracy of 0.1%; low dimensional problems.

	P-algorithm				**EGO**				**osEGO**				**RBF-G**				**CORS-RBF**				**fBcEGO**			
Problem	Fail %	mean	min	max	Fail %	mean	min	max	Fail %	mean	min	max	Fail %	mean	min	max	Fail %	mean	min	max	Fail %	mean	min	max
SCHUBERT	0.00	49.22	37	76	0.00	40.67	30	47	0.00	31.78	12	54	0.00	42.89	33	55	33.33	62.50	36	98	0.00	37.22	32	49
BRANIN	44.44	46.00	33	55	33.33	54.17	45	67	0.00	60.11	36	81	0.00	69.33	42	99	0.00	39.89	25	65	0.00	50.67	35	73
CAMEL3	0.00	45.22	15	82	0.00	20.56	9	31	0.00	41.78	14	64	0.00	35.11	19	61	0.00	21.89	9	31	0.00	21.56	7	30
CAMEL6	0.00	55.00	25	81	0.00	33.22	27	46	0.00	41.33	16	65	0.00	38.22	24	85	0.00	30.22	19	38	0.00	31.22	26	40
DIXONPR	0.00	49.67	27	90	0.00	29.89	21	44	0.00	31.56	22	53	0.00	48.67	38	67	0.00	34.00	20	59	0.00	27.33	21	35
GOLDPR	22.22	68.43	39	95	77.78	55.00	54	56	88.89	42.00	42	42	66.67	67.00	57	83	0.00	40.00	22	61	66.67	43.33	27	58
PAVIANI2	55.56	108.00	75	131	77.78	86.00	64	108	55.56	74.25	31	122	100.00	–	–	–	55.56	80.25	54	110	100.00	–	–	–
SHEKELF	77.78	64.50	58	71	33.33	51.00	18	85	100.00	–	–	–	55.56	71.00	58	90	33.33	47.83	31	94	77.78	66.00	60	72
GULFDR	100.00	–	–	–	100.00	–	–	–	33.33	125.00	88	169	100.00	–	–	–	100.00	–	-1	–	100.00	–	–	–
HARTMAN3	22.22	44.43	23	60	0.00	52.22	37	64	55.56	56.25	32	93	0.00	74.67	29	92	0.00	61.11	43	79	0.00	48.33	32	65

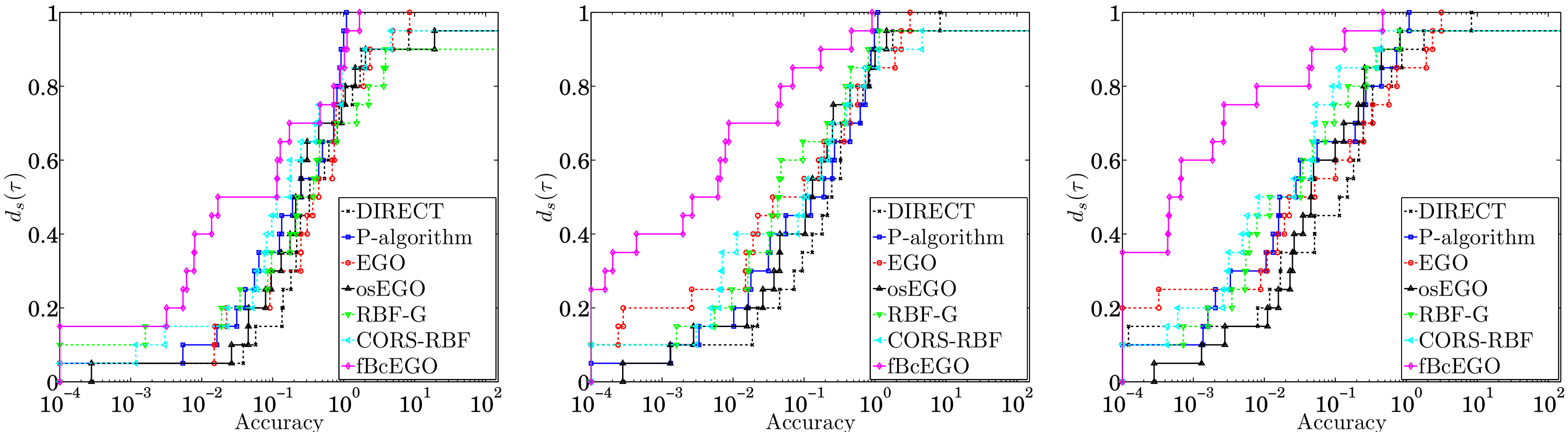
8.4 Experiment 3: Handling Of Nonlinear Constraints

Figure 59 shows the data profiles for the tests on the nonlinearly constrained problems. In contrast with the bound constrained results (Figure 55), `fBcEGO` shows a significant performance gain over its competitors. It was inferred in §4.2.1 that the fully Bayesian approach would result in better performance for nonlinearly constrained problems because the new, more informative modeling approach is applied to all constraint functions. Compare this with the naive likelihood based approaches, which, as discussed in §2.2.4, can fail to provide an adequate representation of a function. This can occur with any number of the constraint functions, which can result in nonsensical sample placement early on due to the multiplicative nature of the constrained EI criterion, equation (4.4). `fBcEGO` solves up to three times as many problems for a given accuracy and simplex gradient budget.

In terms of robustness, i.e., the worst value of ϵ attained by an algorithm in Figure 59, `fBcEGO` surpasses the competing algorithms if the outliers are excluded (see Figure 60). The median accuracy attained by `fBcEGO` is approximately three times better ("half an order of magnitude") than that attained by the closest competitor for $N = 5$, one order of magnitude better for $N = 10$, and one and a half orders of magnitude better for $N = 20$. For $N = 5$, the median of `fBcEGO` is at approximately the same level as or lower than the first quantile of all its competitors. For $N = 10$, the third quantile of `fBcEGO` is lower than the medians of all its competitors. For $N = 20$, the third quantile of `fBcEGO` is approximately equal to the first quantile of its closest competitor. In terms of dominance, `fBcEGO` dominates the competing algorithms for $N = 5$ if the outliers are excluded. For $N = 10$ and $N = 20$, `fBcEGO` dominates even with the outliers. The superiority of `fBcEGO` on this test problem set is clear. The remaining surrogate model-based algorithms exhibit similar median performance while `DIRECT` expectedly performs the worst because it is limited to sampling along coordinate directions.

For the equality constraint violation (Figure 61), the trends are identical. The data profile of `fBcEGO` dominates the competitors with minor exception and if the outliers are excluded(see Figure 62). `fBcEGO` exhibits superior median performance in terms of equality constraint violation. In Figure 62 The median performance of the competing algorithms remains relatively unchanged with increasing N but improves for `fBcEGO`.

Figure 63 shows the performance profiles of the accuracy attained and equality constraint violation for all N condensed into one one data profile for each algorithm. The superiority of `fBcEGO` on nonlinearly constrained problems is clear. The data profile of `fBcEGO` dominates the data profiles of the competitors if all outliers are excluded.

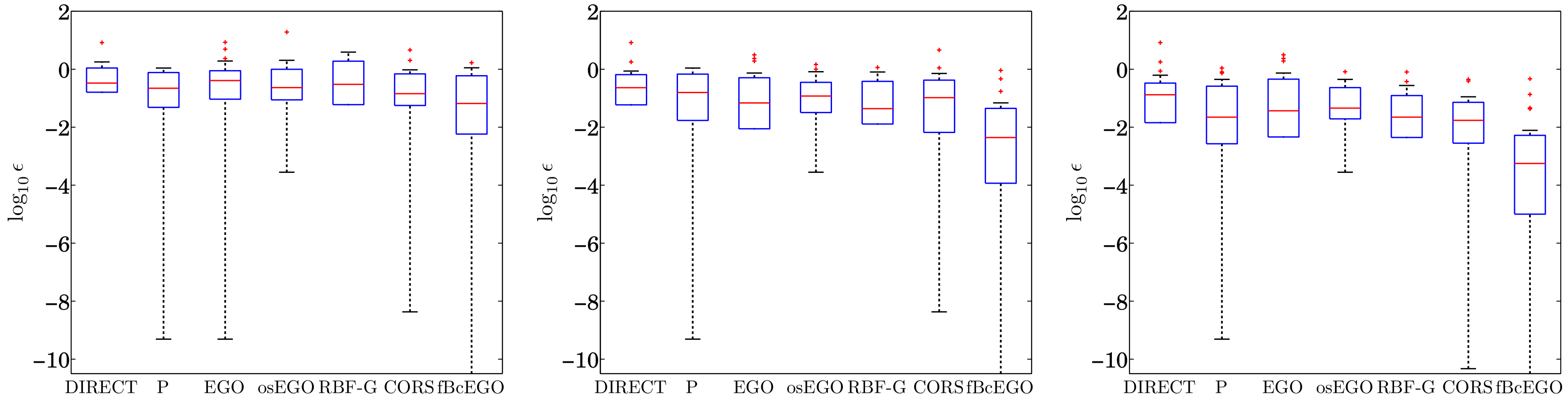


(a) $N = 5$

(b) $N = 10$

(c) $N = 20$

Figure 59: Results of nonlinearly constrained tests; algorithms were executed for a maximum of N = 5, 10, and 20 simplex gradients and the accuracy attained was recorded.

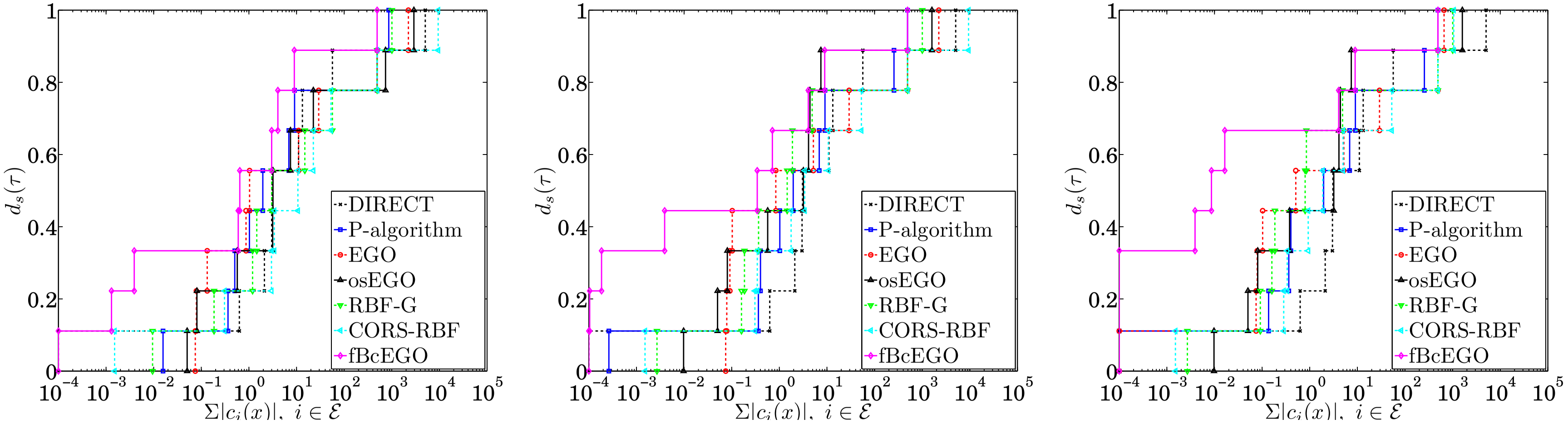


(a) $N = 5$

(b) $N = 10$

(c) $N = 20$

Figure 60: Box plots of nonlinearly constrained tests with accuracy metric as a function of budgeted simplex gradients N.

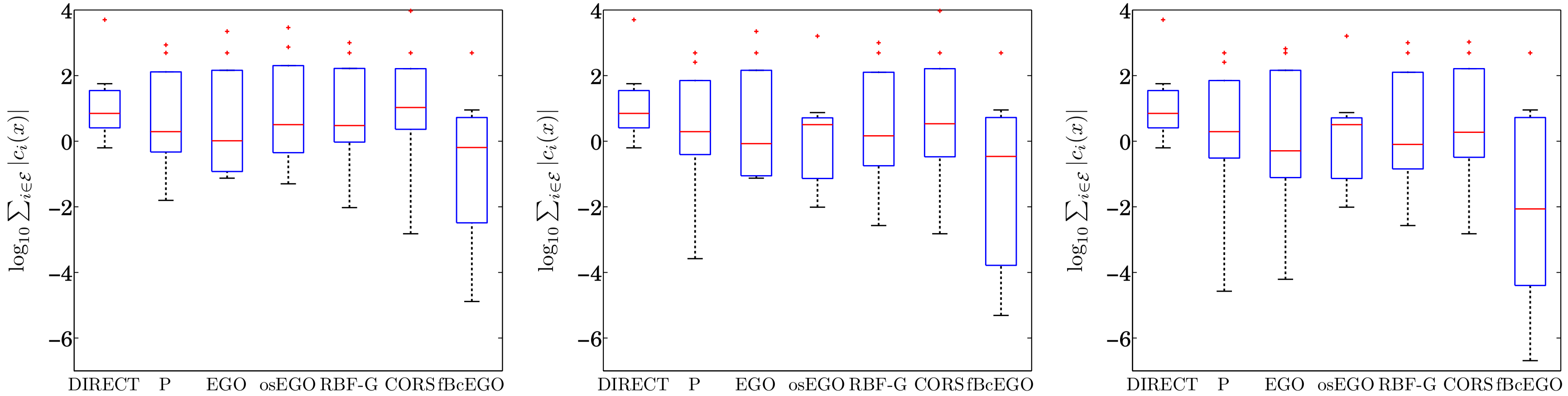


(a) $N = 5$

(b) $N = 10$

(c) $N = 20$

Figure 61: Results of nonlinearly constrained tests; algorithms were executed for a maximum of 5, 10, and 20 simplex gradients and the equality constraint violation was recorded.



(a) $N = 5$

(b) $N = 10$

(c) $N = 20$

Figure 62: Box plots of nonlinearly constrained tests with equality constraint violation as a function of budgeted simplex gradients N.

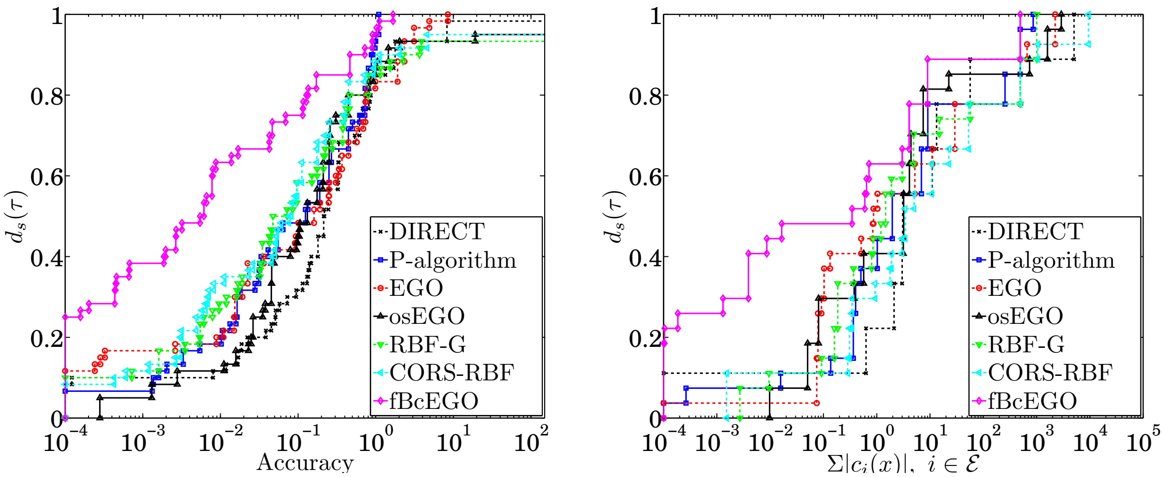


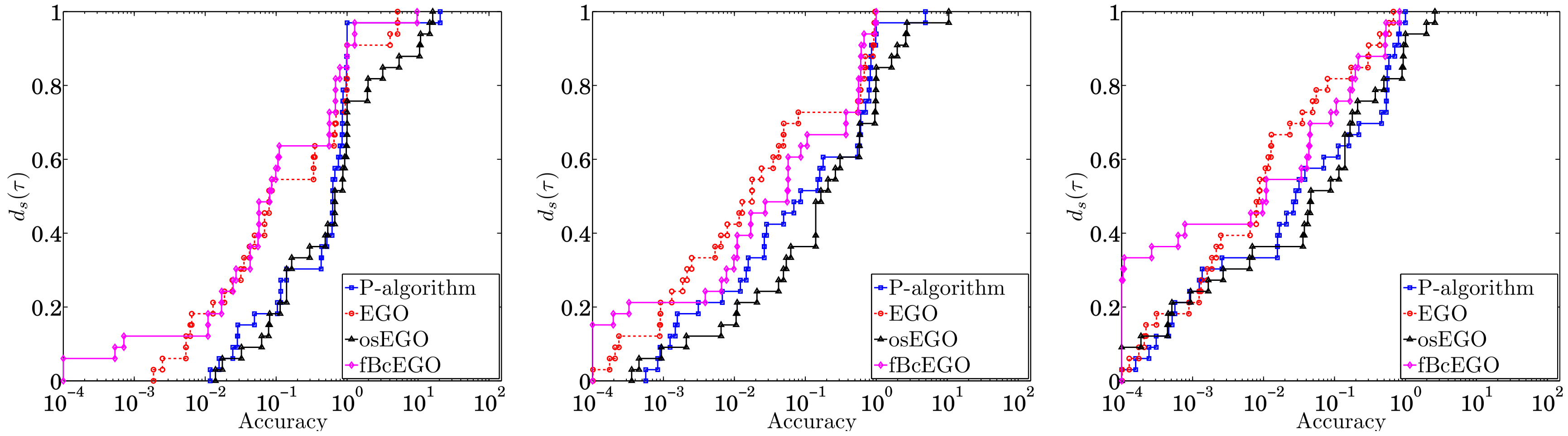
Figure 63: Results of nonlinearly constrained tests for all N. Accuracy, left; equality constraint violation, right.

8.5 Experiment 4: Noise Corrupted Observations

As discussed in §5.4, an additional GP hyperparameter is required to model output-dependent noise. For the likelihood-based algorithms P-algorithm and EGO, this means that the MLE problem (2.15) is two-dimensional; for osEGO this problem is $n+2$ dimensional. For the fully Bayesian approach, a joint prior $p(\theta, \theta_n)$ must be placed over θ and θ_n. The algorithms have no knowledge of whether or not a function being minimized is truly noise-free. Thus, when regression algorithms are applied to noise-free problems, the resulting performance will be fundamentally different from the performance of interpolative algorithms applied to the same problems. This is taken into account in the test matrix for noise corrupted problems (Figure 53).

The competing algorithms for the noise corrupted problems are listed in Table 14 and the test problems are those listed in Table 18 but corrupted with output-dependent noise as discussed in §5.4. Experiments are conducted for the three noise levels $\epsilon_n = 0\%, 0.01\%$, and 1% with three budgets of $N = 5$, 10, and 20 simplex gradients. The data profiles and box plots for each N are given in Figures 64 and 65, respectively. Each panel in Figure 64 includes all test problems over all three noise levels, treating each noise level as a different test. The performance profile of fBcEGO most closely competes with that of EGO, with fBcEGO able to solve more problems to high accuracies (lower ϵ). The data profiles of EGO and fBcEGO dominate those of osEGO and the P-algorithm, with minor exception. Excluding outliers, the P-algorithm, EGO, and fBcEGO exhibit similar robustness for all N. In terms of median performance, EGO and fBcEGO show similar median accuracy attained as well as a similar variation in the performance. Both algorithms exhibit superior median performance when compared with the P-algorithm and osEGO.

Figure 66 shows the results as a function of N and ϵ_n. An important observation from this chart is that fBcEGO can attain a higher accuracy (lower ϵ) on a significantly larger percentage of problems than any other algorithm. For example, for the case $(N, \epsilon) = (20, 0.01)$, fBcEGO can solve approximately 30% of the problems to within an accuracy of $\epsilon \leq 10^{-4}$. The ability of fBcEGO to recover the true global minimum to such high accuracies in the presence of noise can be attributed to the fully Bayesian approach. By fitting many different models and judging them from a Bayesian viewpoint, more can be learned about the hyperparameters. This property may prove useful when the algorithm is applied to real-life design problems.

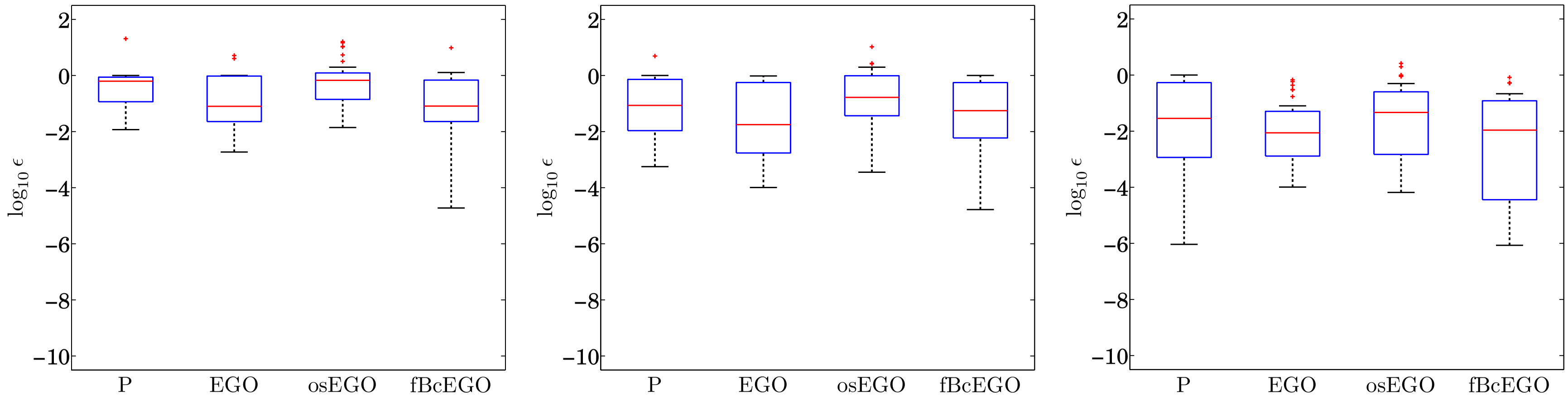


(a) $N = 5$

(b) $N = 10$

(c) $N = 20$

Figure 64: Results of noise corrupted tests; algorithms were executed for a maximum of $N = 5$, 10, and 20 simplex gradients and the accuracy attained was recorded. For each N, three noise levels are included in the results: $\epsilon_n = 0\%$, 0.01%, and 1%.



(a) $N = 5$

(b) $N = 10$

(c) $N = 20$

Figure 65: Box plots of noise corrupted tests with accuracy metric versus simplex gradient budget N. For each N, three noise levels are included in the results: $\epsilon_n = 0\%$, 0.01%, and 1%.

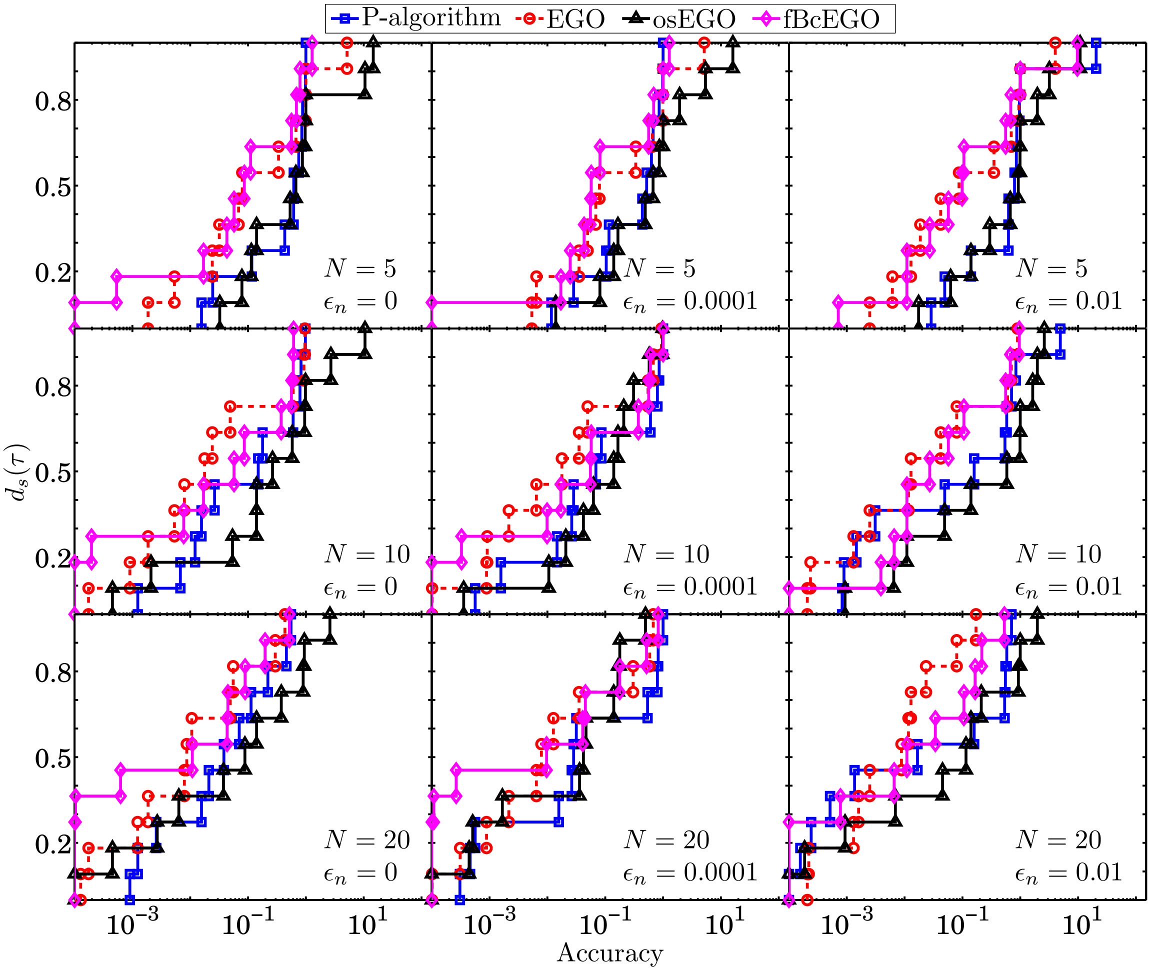


Figure 66: Results of noise corrupted tests by simplex gradient budget N and noise level ϵ_n; noise level displayed as an absolute quantity.

8.6 Experiment 5: Nonsmooth Problems

It was discussed in §2.2 that different basis functions may lead to improved modeling of certain characteristics, which can improve the performance of surrogate model-based global optimization algorithms on specific problems. While experiments were found [104] that assess the global modeling accuracy of GPs with different covariance functions and fitting methods, no experiments were found that assess the performance of global optimization algorithms with different basis functions. This is not a trivial difference, because the goal of global optimization algorithms is not to develop a globally accurate surrogate model of the design space, but rather to focus the limited function evaluations in promising areas where the global minimum is likely to occur.

This section presents the performance of the nonsmooth algorithms listed in Table 15 on the nonsmooth problems listed in Table 10. An important distinction for these problems is that the test functions are nondifferentiable at the global minimizer. The tests would be irrelevant if the minimizer occurred in the differentiable region. An important potential application of nonsmooth basis functions, besides solving nonsmooth problems, is in the solution of constrained problems where a nonsmooth penalty function, e.g., equation (4.1), is used and there is at least one active constraint. In such cases, the transformed problem will be nondifferentiable at the global minimizer.

Figure 67 shows the data profiles of the number of simplex gradients N required to attain an accuracy of 1% over the nine initial designs from Table 5. For nonsmooth basis functions, `fBcEGO` dominates the competing algorithms, with minor exception. `fBcEGO` also exhibits no failures and is the most reliable, solving 100% of the problems at the least cost. For smooth basis functions (Figure 67, left), CORS-RBF dominates the competing algorithms, with minor exception. Using a smooth basis function to solve nonsmooth functions results in significantly worse performance when compared with the performance that can be obtained by employing nonsmooth basis functions for the same problems. This is due to the inability of the smooth basis functions to accurately model nondifferentiable subspaces. It is emphasized that the test problems were nondifferentiable at the global minimizers. It is inferred that if the global minimizers occurred in differentiable regions, the smooth basis functions would have outperformed the nonsmooth basis functions.

Figure 68 shows the box plots of the performance. Comparing the box plots for smooth basis functions (left) to those of nonsmooth basis functions (right), it can be seen that incorporating the proper prior knowledge about the problem being solved can lead to substantial performance improvements. For example, compare the box plot corresponding to `fBcEGO` for smooth and nonsmooth basis functions to see that the median performance has improved by approximately 100% while the interquartile range has also been reduced by a factor of two. An unexpected result is that CORS-RBF performed better on nonsmooth problems with a smooth basis function. CORS is the only algorithm that uses the surrogate model directly as the ISC, thus the accuracy of the surrogate model becomes a factor in the performance. Since the nondifferentiable subspaces of the nonsmooth problems are relatively small, using the smooth basis function resulted in better performance.

It remains to examine the performance of nonsmooth algorithms on smooth problems (Experiment 5.3). The nonsmooth algorithms are applied to the smooth problems in Table 18 over nine initial designs and the cost in simplex gradients N required to attain 1% accuracy is recorded. Indeed, the general performance trend is negative for algorithms with nonsmooth basis functions, as they are unable to model curvature to the same degree as the smooth basis functions. The only exception is `fBcEGO`, which shows identical performance up to $N = 15$ regardless of basis function. This atypical behavior of `fBcEGO` is a manifestation of the fully Bayesian approach, which constructs a finite mixture model, thereby recovering (to a degree) the curvature-modeling abilities of the smooth basis functions.

Problem specific analyses of the tests performed in this section are given in Tables 21 – 23. Table 21 shows the problem specific analysis for the smooth algorithms on the nonsmooth problems. The most difficult problems for the smooth algorithms were NS2D4, NS2D5, and NS2D7. These problems had global minimizers in steep nondifferentiable "valleys". The performance on these problems improves dramatically when nonsmooth algorithms are used (Table 22). Finally, Table 23 shows the performance of the nonsmooth algorithms on smooth problems. The difficult problems in this case are the Paviani problem and Shekel's Foxholes.

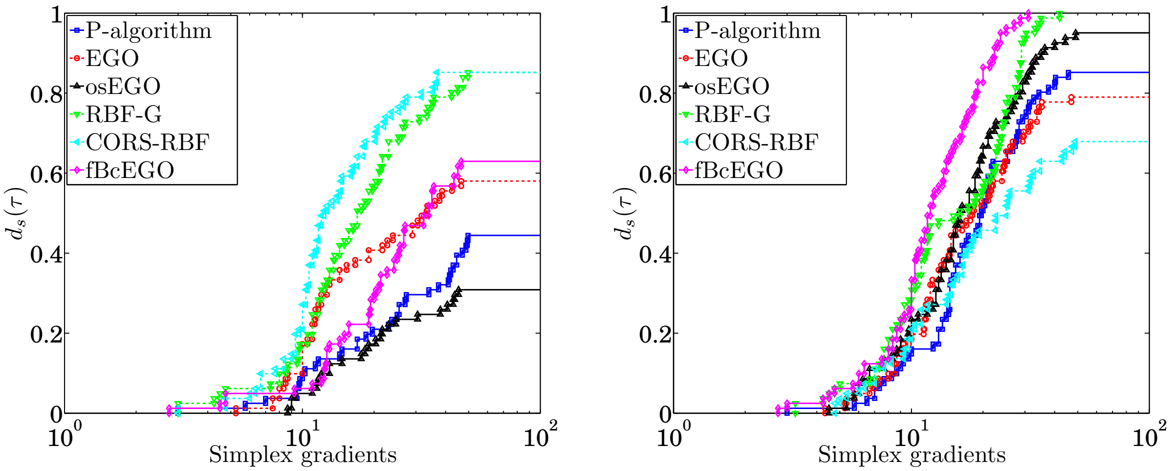


Figure 67: Performance of competing algorithms on nine nonsmooth problems with $n \leq 3$ over nine initial designs; percentage of problems solved as a function of simplex gradients required to achieve $\epsilon \leq 1\%$. Smooth basis functions (left) and nonsmooth basis functions (right) from Table 15.

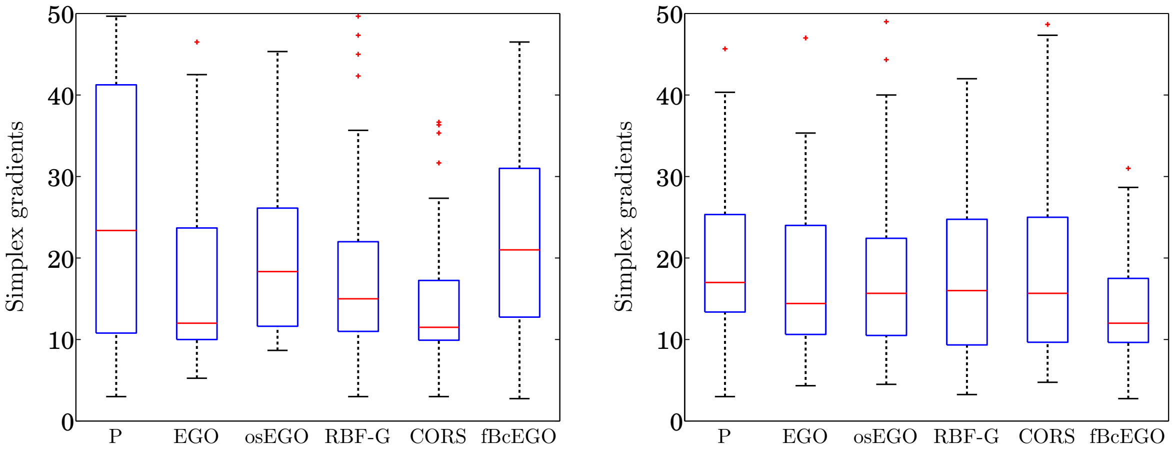


Figure 68: Box plots of test results of nine nonsmooth problems with $n \leq 3$ over nine initial designs; successful cases only. Smooth basis functions (left) and nonsmooth basis functions (right) from Table 15.

Table 21: Number of function evaluations required by smooth algorithms to achieve an accuracy of 1% on nonsmooth problems.

	P-algorithm				**EGO**				**osEGO**				**RBF-G**				**CORS-RBF**				**fBcEGO**			
Problem	Fail %	mean	min	max	Fail %	mean	min	max	Fail %	mean	min	max	Fail %	mean	min	max	Fail %	mean	min	max	Fail %	mean	min	max
NS1D	22.22	63.71	20	94	0.00	40.33	15	93	66.67	37.67	26	61	0.00	28.89	19	46	0.00	24.22	20	29	11.11	61.00	24	93
NS2D1	55.56	94.75	43	148	33.33	60.00	24	108	55.56	62.25	27	129	0.00	54.33	24	100	0.00	29.22	20	40	22.22	70.29	37	130
NS2D2	66.67	117.67	81	149	44.44	78.40	30	116	66.67	90.33	61	136	0.00	39.00	29	64	0.00	30.44	20	35	33.33	79.17	43	130
NS2D3	77.78	90.00	34	146	100.00	–	–	–	100.00	–	–	–	33.33	109.17	79	149	66.67	83.67	32	110	66.67	109.00	57	135
NS2D4	100.00	–	–	–	100.00	–	–	–	100.00	–	–	–	66.67	97.67	65	135	22.22	81.86	54	110	100.00	–	–	–
NS2D5	66.67	28.33	21	35	0.00	32.22	26	38	55.56	32.50	26	36	0.00	35.33	22	47	0.00	27.00	18	35	0.00	46.56	37	64
NS2D6	11.11	82.63	44	126	0.00	53.67	34	100	11.11	68.50	27	122	0.00	62.78	35	107	0.00	54.11	28	82	0.00	71.44	33	107
NS2D7	100.00	–	–	–	100.00	–	–	–	77.78	94.00	57	131	33.33	80.50	35	142	0.00	47.22	25	76	100.00	–	–	–
NS3D	0.00	54.44	12	102	0.00	61.67	21	122	88.89	46.00	46	46	0.00	31.22	12	76	44.44	34.60	12	81	0.00	53.00	11	102

Table 22: Number of function evaluations required by nonsmooth algorithms to achieve an accuracy of 1% on nonsmooth problems.

	P-algorithm				**EGO**				**osEGO**				**RBF-G**				**CORS-RBF**				**fBcEGO**			
Problem	Fail %	mean	min	max	Fail %	mean	min	max	Fail %	mean	min	max	Fail %	mean	min	max	Fail %	mean	min	max	Fail %	mean	min	max
NS1D	11.11	32.63	27	42	0.00	24.22	17	34	0.00	39.11	28	57	0.00	41.56	20	50	0.00	46.89	33	63	0.00	21.89	15	28
NS2D1	66.67	35.67	21	49	22.22	60.86	24	106	0.00	35.33	20	49	0.00	26.67	15	45	22.22	32.86	21	44	0.00	30.78	24	41
NS2D2	11.11	43.13	27	65	22.22	57.43	36	80	0.00	41.56	16	82	0.00	30.56	21	39	11.11	46.63	21	58	0.00	39.56	27	56
NS2D3	0.00	87.11	43	121	44.44	71.60	41	101	11.11	88.75	75	105	0.00	94.67	77	126	100.00	–	–	–	0.00	47.67	25	60
NS2D4	44.44	74.20	46	120	77.78	103.50	66	141	22.22	103.29	58	147	0.00	75.78	48	91	88.89	68.00	68	68	0.00	53.78	35	67
NS2D5	0.00	40.56	21	76	0.00	29.67	13	59	0.00	30.00	16	47	0.00	29.67	22	36	0.00	32.78	18	53	0.00	22.67	17	31
NS2D6	0.00	82.67	56	137	11.11	80.00	44	103	0.00	55.22	39	68	0.00	66.67	23	87	0.00	105.22	16	146	0.00	65.11	32	93
NS2D7	0.00	63.67	42	83	11.11	50.38	27	95	0.00	50.67	23	88	0.00	62.22	52	73	66.67	79.67	51	95	0.00	51.00	31	71
NS3D	0.00	38.44	12	60	0.00	35.44	21	49	11.11	44.00	18	129	0.00	27.11	13	47	0.00	31.89	19	47	0.00	26.44	11	48

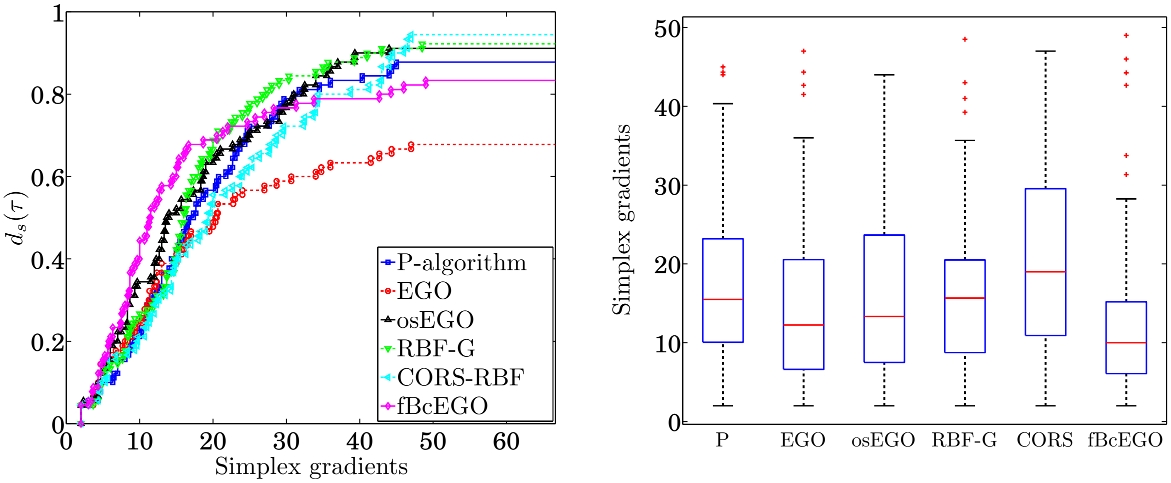


Figure 69: Performance of nonsmooth algorithms on ten smooth problems with $n \leq 3$ over nine initial designs; percentage of problems solved as a function of simplex gradients required to achieve $\epsilon \leq 1\%$ (left) and box plots of successful cases (right).

Table 23: Number of function evaluations required by nonsmooth algorithms to achieve an accuracy of 1% on smooth problems.

	P-algorithm				**EGO**				**osEGO**				**CORS-RBF**				**RBF-G**				**fBcEGO**			
Problem	Fail %	mean	min	max	Fail %	mean	min	max	Fail %	mean	min	max	Fail %	mean	min	max	Fail %	mean	min	max	Fail %	mean	min	max
SCHUBERT	22.22	42.00	14	58	88.89	26.00	26	26	0.00	48.00	12	88	0.00	78.89	57	94	0.00	51.89	32	97	44.44	12.20	7	17
BRANIN	0.00	47.56	34	61	11.11	78.75	16	141	22.22	73.00	39	111	0.00	56.00	32	79	0.00	47.56	24	62	22.22	66.00	38	94
CAMEL3	0.00	32.89	9	67	22.22	38.71	11	92	0.00	15.11	7	25	0.00	19.78	9	33	0.00	18.00	11	25	0.00	15.67	9	27
CAMEL6	0.00	77.89	32	135	11.11	57.88	14	133	0.00	28.11	16	40	0.00	40.56	17	59	0.00	38.56	14	60	0.00	26.89	17	37
DIXONPR	44.44	24.20	14	39	22.22	31.86	14	69	0.00	39.78	14	57	0.00	58.56	14	103	0.00	57.89	14	123	0.00	28.78	14	50
GOLDPR	33.33	54.00	28	132	44.44	30.20	23	34	0.00	52.00	26	90	0.00	54.00	36	86	0.00	44.44	33	50	0.00	50.89	19	138
PAVIANI2	22.22	32.71	6	89	33.33	24.67	6	62	44.44	22.20	6	87	0.00	50.56	6	132	22.22	43.57	6	107	33.33	19.17	6	46
SHEKELF	0.00	62.56	30	108	55.56	34.75	25	45	0.00	74.33	25	118	55.56	58.50	25	97	44.44	67.80	26	129	55.56	41.00	30	56
GULFDR	0.00	94.33	41	138	11.11	91.88	49	166	22.22	100.00	54	157	0.00	144.78	73	184	11.11	89.38	53	157	0.00	95.00	46	196
HARTMAN3	0.00	50.00	26	82	22.22	37.86	15	63	0.00	43.33	22	56	0.00	77.67	27	172	0.00	54.44	21	99	11.11	56.13	21	177

8.7 Experiment 6: Hard Constraints

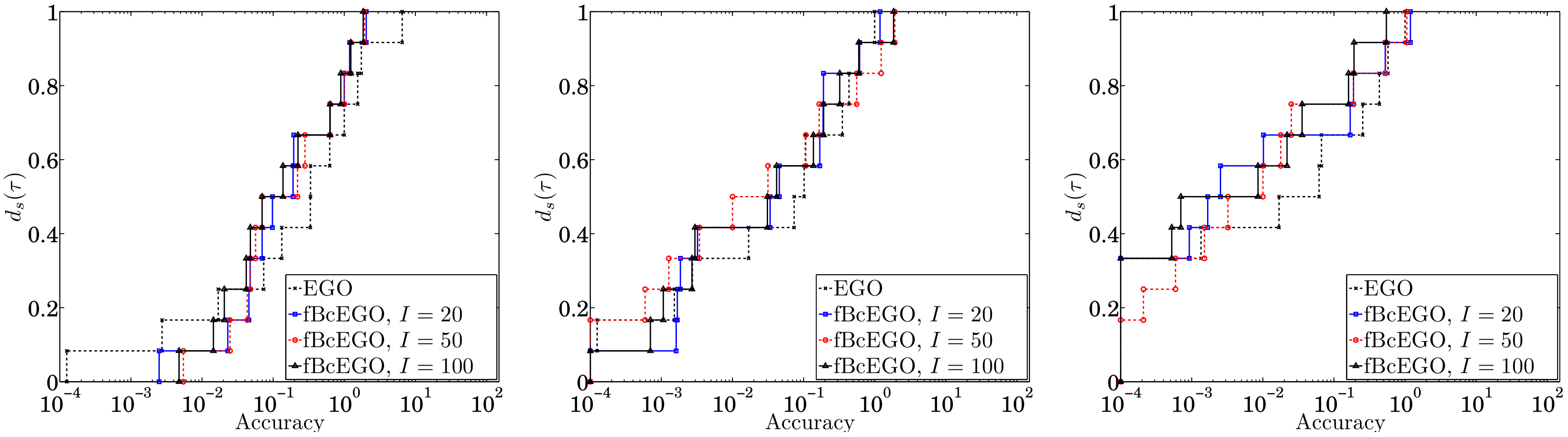
See Chapter 9.

8.8 Additional Findings

8.8.1 Effect Of Prior Density On The Performance Of fBcEGO

fBcEGO uses $I = 100$, i.e., 100 elements in $p(\theta)$, with $-5 \leq \log\theta \leq \log(1.5)$ and θ uniformly distributed in the log-space as an initial assumption. The model fitting method in §2.2.6 then allows small changes in the number of elements while simultaneously expanding the domain of θ if necessary before cropping the domain to only include 99.99% of the largest contributors to $p(\theta|y^{(k)})$. The posterior is then recomputed and used to construct the GP model. In many noise-free cases, this process resulted in a small number of important modes, say 5–10. A user may desire to know the consequence of starting out with a less populated domain for θ, say, $I = 20$ or $I = 50$, since this would reduce the analytical effort of fBcEGO.

An experiment is conducted to assess the performance of fBcEGO with $I = 20$, 50, and 100 over $-5 \leq \log\theta \leq \log(1.5)$ on low dimensional problems of varying difficulty (problems B-1, B-2 through B-6, B-8, B-10, and B-13; problems B-7 and B-9 with $n = 2$; and B-11 with $n = 3$). Figure 70 shows the accuracy attained after $N = 5$, 10, and 20, with the MLE-based algorithm EGO included as the baseline method. The corresponding box plots are shown in Figure 71. While the median performance and spread of fBcEGO for all I is better than the median performance of EGO, no definite improvement can be ascertained between the three cases of I. It is inferred that for noisy problems which require a two dimensional prior $p(\theta, \theta_n)$, a reduced value of I can be used to obtain similar performance to a highly populated prior $p(\theta)$ but at a lower analytical cost.

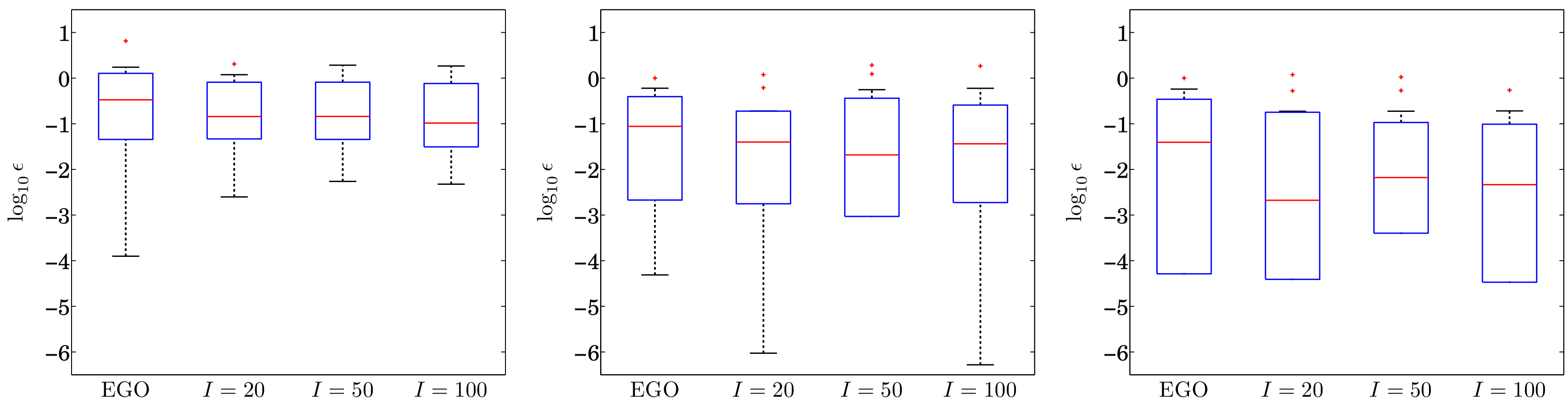


(a) $N = 5$

(b) $N = 10$

(c) $N = 20$

Figure 70: Performance of `fBcEGO` with varying I on eleven bound constrained problems; algorithms were executed for a maximum of $N = 5$, 10, and 20 simplex gradients and the accuracy attained was recorded.



(a) $N = 5$

(b) $N = 10$

(c) $N = 20$

Figure 71: Box plots of the performance of `fBcEGO` with varying I on eleven bound constrained problems with accuracy metric as a function of budgeted simplex gradients N.

CHAPTER IX

APPLICATIONS

The global optimization methodology is applied to two aircraft-related design problems. The first problem deals with the design of an airfoil section for minimum drag. The second problem deals with the conceptual design of a commercial aircraft from an existing baseline with the goal of minimizing the maximum gross takeoff weight under performance constraints. These problems will serve to evaluate the capabilities of the methodology with respect to real-life design problems.

Recall that the motivation for this research (§1.1) was the need for an efficient global optimization algorithm for unconventional designs or novel concepts. The design codes involved were assumed to be computationally expensive and subject to the technical challenges listed in §1.2. The application problems in this chapter are neither expensive nor are they unconventional in the sense that a non-derivative design is being produced. Remark 7.3.1 discussed why the evaluation time of the test problems is irrelevant for testing purposes. The application problems exhibit the technical challenges listed §1.2, which are properties of expensive design codes describing unconventional vehicles. Thus, the fact that the designs are not unconventional does not preclude their use as a means to evaluate the primary hypothesis and research objective. The familiar solutions obtained in the application problems may be better suited than unconventional designs to illustrate the capabilities of `fBcEGO`.

9.1 Airfoil Section Design

This problem was inspired by [38].

9.1.1 Background & Challenges

The combination of optimization algorithms and CFD solvers offers promise for the development of improved aerodynamic designs. In this application, an airfoil based on the supercritical NASA SC(2)-0610 airfoil [47] is optimized for sectional drag c_d. Analysis is provided by XFOIL v6.94 [31], a linear-vorticity stream function panel method for subsonic calculations, and optimization is performed using the methodology in Chapter 6.

There are two main difficulties in airfoil section design. The first is the parameterization of the airfoil section such that various shapes can be explored efficiently. Two requirements guide the selection of the parameterization scheme: minimization of design variables and orthogonality of design functions. Airfoils are represented as a set of upper and lower coordinates z^u and z^ℓ for the upper and lower surfaces, respectively, at the same chordwise locations x. The requirements can be satisfied by employing orthogonal basis functions to represent the airfoil rather than using the coordinates of the airfoil section as the design variables. Orthogonality allows exploration of all possible airfoil sections that are spanned by the basis functions through a linear combination of the basis functions. Lack of orthogonality implies a nonunique mapping of the parameter values to the geometry. The resulting spurious multimodality of the objective function can degrade the search process [57].

A recent method to generate orthogonal basis functions for an airfoil is to represent the airfoil using a Savitzky-Golay smoothing filter (call this function f_1) and to then generate subsequent functions by applying the smoothing filter to the residuals of the preceding fit [101, 38, 90]. Each basis function is characterized as a set of upper and lower z values with chordwise distribution matching the original coordinates. For this problem, the chordwise locations are defined by the cosine spacing $x/c = \frac{1}{2}(1 - \cos\beta)$ where $0 \leq \beta \leq \pi$ and z values were obtained by cubic spline interpolation of the original linearly spaced coordinates [47]. This gave significantly improved

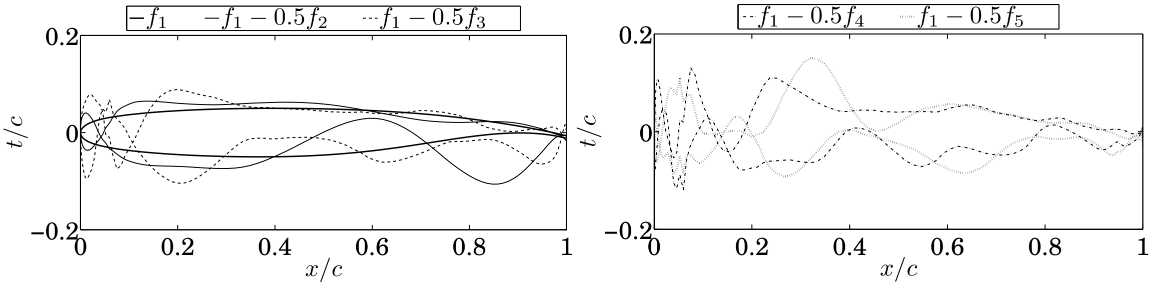
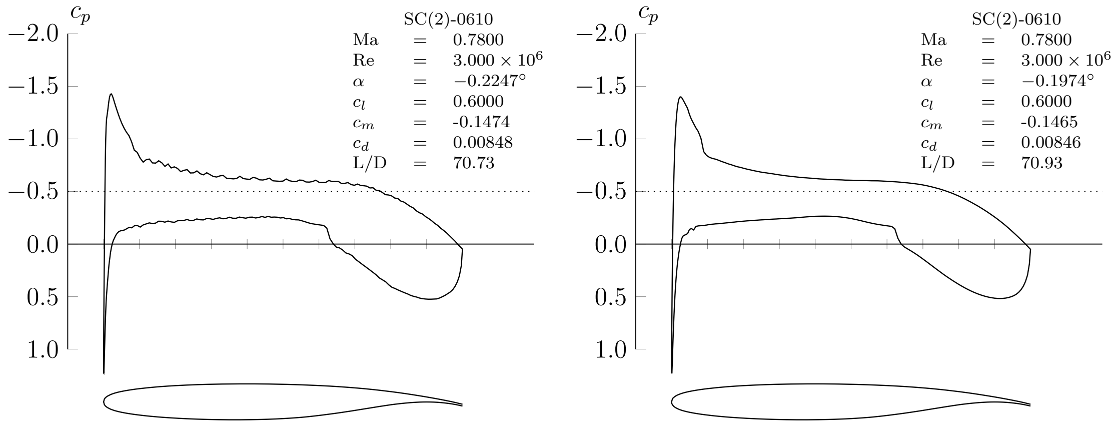


Figure 72: Orthogonal basis function airfoil parameterization for NASA SC(2)-0610 airfoil.

representation at the leading and trailing edges and a smoothed pressure distribution, which led to a sectional drag value of $10^4 \times c_d = 84.6$ for f_1 as computed by XFOIL (compare Figure 73a with Figure 73b). The original SC(2)-0610 airfoil designed by a hodograph method is reported to have a sectional drag coefficient of $10^4 \times c_d = 84.7$. The drag coefficient of the original airfoil as computed by XFOIL is $10^4 \times c_d = 84.8$. The residual functions f_2, f_3, f_4, and f_5 are orthogonalized via singular value decomposition and then added to f_1 with weightings w_2, w_3, w_4, and w_5, respectively. The equality constraints $c_l = 0.6$ and $(t/c)_{\max} = 10\%$ are enforced internally such that the parameterized airfoil remains in the SC(2)-0610 class. The first five basis functions are shown in Figure 72. Nonsensical airfoils are produced by adding individual functions with large weightings, but a high degree of geometry control can be achieved by combining the functions and optimizing their weightings. A technical challenge of this parameterization approach is that bounds on the design variables w_i are unknown and non-intuitive and must be obtained from previous knowledge or by some inexpensive application-specific method. Thus, while the airfoil is not necessarily an unconventional design, the design space in terms of the weightings w_i is unfamiliar.

The second difficulty with airfoil section design involves the analysis code that is



(a) Original airfoil

(b) Baseline airfoil f_1

Figure 73: Surface pressure distributions.

used, XFOIL in this case. XFOIL is a subsonic analysis code which will not converge when the local speed over the airfoil becomes too large, when the boundary layers separate, or when a nonsensical airfoil is analyzed. Analysis of nonsensical airfoils can be avoided by checking the coordinates, but the other two failure modes cannot be predicted. Thus, there will be hard constraints in this application. Discretization error and rounding in the analysis code also makes the problem noisy (XFOIL only reports c_d values to five decimal places). In some cases, XFOIL may not converge unless the solution is initiated from another converged solution. When this safeguard fails, "holes," i.e., missing data, will occur in the design space.

9.1.2 Competing Methods

Besides `fBcEGO`, two state-of-the-art methods are considered: a genetic algorithm (GA) and `DIRECT`, both of which have been modified to handle hard constraints. The GA is a canonical binary-coded implementation, e.g., [117], using 52 bits of precision on the unit hypercube and a roulette wheel with elitism selection scheme to create subsequent generations. The initial population is randomly generated until $5n$ feasible parents have been created. The standard crossover and mutation operators have been modified to cope with offspring fitness evaluation failures by allowing the parents to

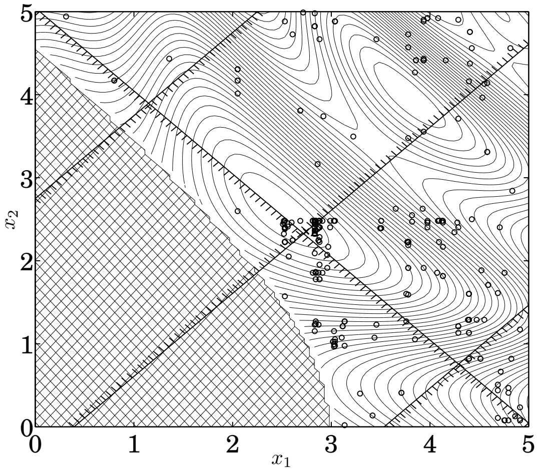


Figure 74: Performance of genetic algorithm modified to handle hard constraints; example nonlinearly constrained problem. Hatched area represents infeasible area by hard constraint.

survive in such cases. An example of the performance on a test function for 150 function evaluations is shown in Figure 74. This problem exhibits two inequality constraints indicated by the hatched lines and one hard constraint indicated by the double-hatched area below the main diagonal of the figure. `DIRECT` with the maximum value imputation method is also included in the set of competing algorithms for this problem (see §4.3).

9.1.3 Methodology Applied To Two-Variable Optimization

9.1.3.1 Step 1: Formal Problem Statement

A two-variable optimization is performed first with the weightings $w_2 \in [-0.1, 0.1]$ and $w_3 \in [-0.04, 0.03]$ and $w_4 = w_5 = 0$. The bounds for the weightings were unknown due to the non-intuitive nature of the design space, i.e., it was difficult to tell what the proper bounds should be given Figure 72. However, the airfoil geometry as a function of w_2 and w_3 can be manually inspected and used to obtain an outer approximation to the feasible area. Figure 75 plots a coarse grid of some of the airfoil shapes that can expected for the given bounds. The hatched area indicates nonsensical airfoils, i.e., $z^\ell > z^u$ for some x. Airfoils in white rectangles indicate failed (`NaN`) values for c_d, and those in shaded rectangles represent successful cases, with the shading becoming darker for lower values of c_d. The airfoil in the square $w_2 = w_3 = 0$ corresponds to the baseline airfoil f_1.

The sectional drag coefficient c_d is to be minimized at $c_\ell = 0.6$, $M_\infty = 0.78$, and $Re = 3 \cdot 10^6$. The thickness-to-chord ratio t/c of the airfoil is fixed at 10% as defined by the last two digits of the four digit NACA code. These constraints are enforced internally. Thus, the problem is only subject to bound constraints and hard constraints. The sectional drag coefficient c_d is then a function of w_2 and w_3 and the "true" function $c_d(w_2, w_3)$ generated by 51×51 simulations on a uniform grid is shown in Figure 76 as a color surface.

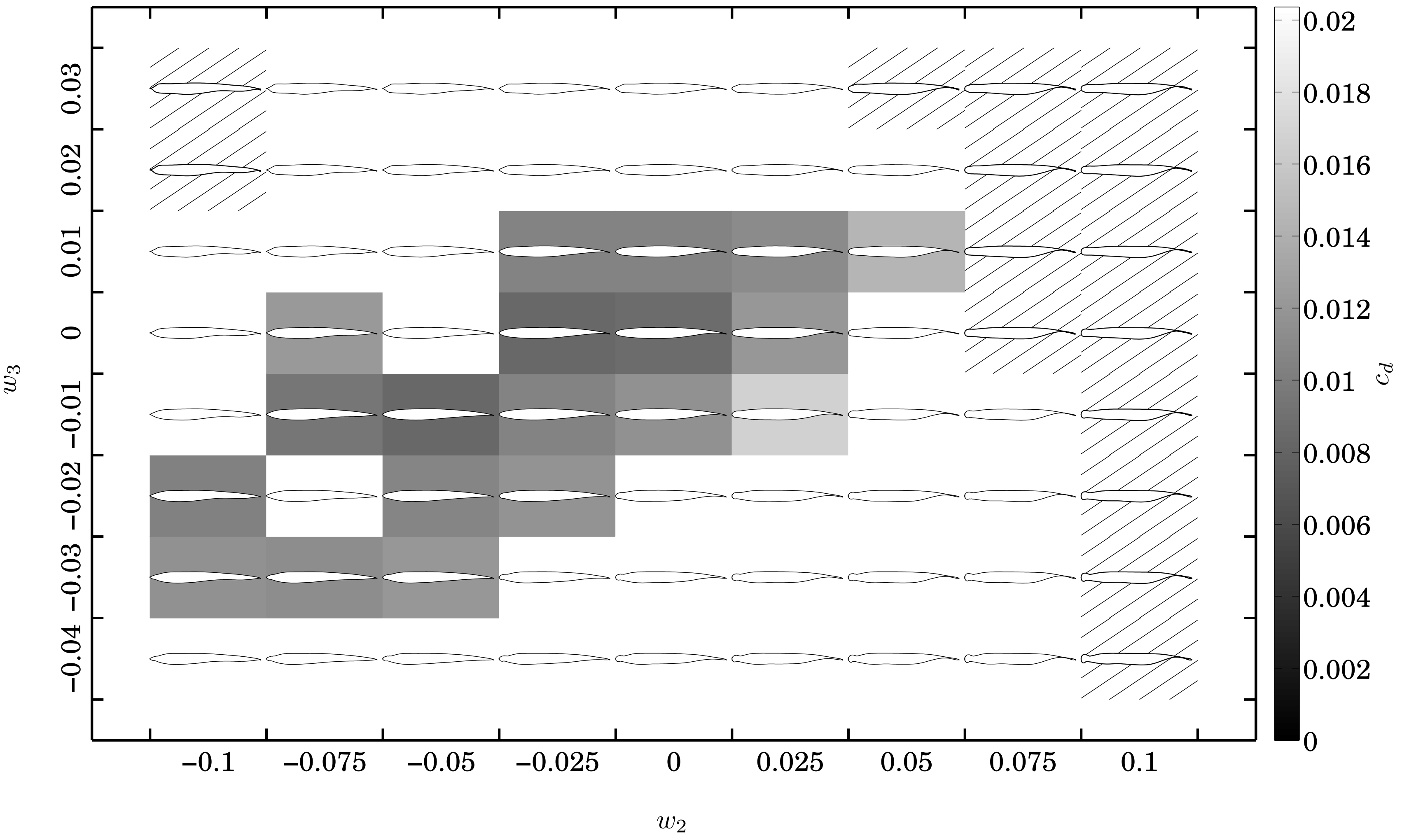


Figure 75: Some possible airfoils that could be generated for the two-dimensional problem. Regions of infeasibility due to nonsensical airfoil geometries are shown as hatched areas.

9.1.3.2 Step 2: Select Initial Design

This problem had hard constraints, but as long as some method is used to assign values to failed iterates, the results from §8.2 can be directly applied. Two designs are selected for this problem: N_1 DGS+FCP and N_2 LHD+FCP. The DGS base design is chosen due to its success in §8.2 and the corner points are added to drive the search away from these areas. `DIRECT` is executed until $N_1 = 6$ *feasible* DGS points are found for the initial design and then four corner points are added. The N_2 LHD base design is chosen for its good performance when the simplex gradient budget is not severely restricted (see Figure 54). Because this problem is expected to produce many failures in the initial design, it may be worthwhile to sample the design space in a space-filling manner such that `fBcEGO` can generate a more accurate initial model. This may help increase the number of successful subsequent iterates due to the improved values generated by the selected imputation method.

The N_2 LHD+FCP initial design returns sixteen failures shown as red spheres in Figure 76 and nine successful function evaluations shown as black spheres, a 64.00% failure rate. In comparison, the failure rate of the 51 × 51 surface was 64.52%. The region of feasibility is well defined, with infeasible areas corresponding to low values of w_2 with high values of w_3, and high values of w_2 with high values of w_3. Failures are also present interior to the feasible region, shown as “holes,” which adds to the difficulty of the problem.

Figure 76 also shows the outcome of applying the penalized imputation scheme from §5.3 to handle hard constraints. A surface (coarse mesh) is fit through the nine initial feasible (black) points, which is then used to impute the values of the failed (red) points using the relation $y(x^{(f)}) \leftarrow \hat{Y}(x^{(f)}) + s(x^{(f)})$. A second surface (fine mesh) is fit through all points which, in combination with the reinterpolation scheme from §5.4, is used to compute the expected improvement. Note that the fine mesh will in general rest above the coarse mesh due to the penalization, but this is not guaranteed. The expected improvement is shown as the filled contour plot on the horizontal plane of Figure 76, with values below $\log \mathbb{E}_I(x) = -11$ omitted to avoid cluttering the figure. The expected improvement is maximized to obtain the next iterate and the process repeats. The situation after three function evaluations (one simplex gradient) is shown in Figure 76 with all three update points being feasible (green spheres). The global minimum value is $10^4 \times c_d = 77.6$ and the global minimizer is indicated on the horizontal plane by the $\times$ symbol.

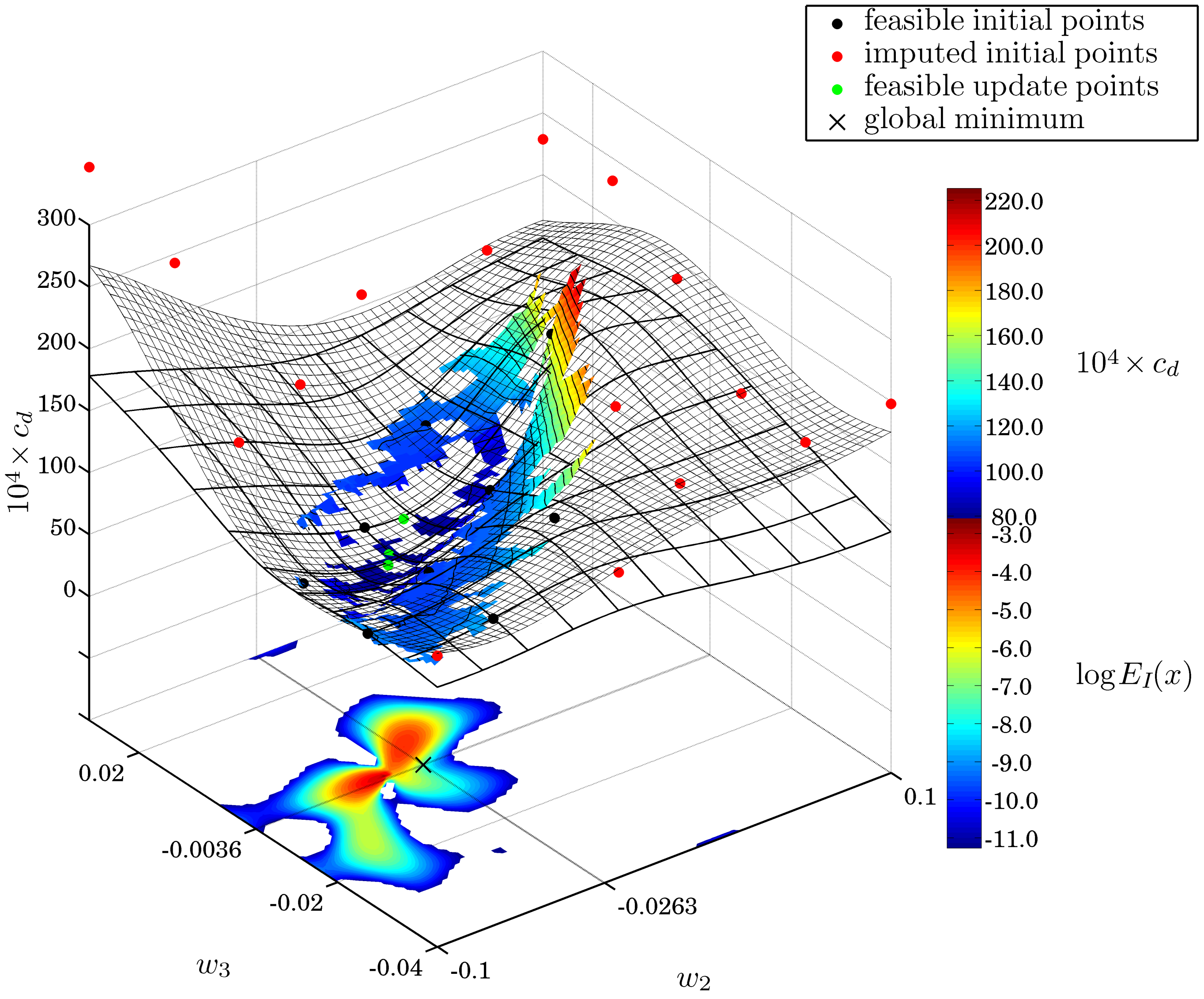


Figure 76: Airfoil optimization design space in two dimensions. True function (colored surface), prediction based on feasible points (coarse mesh), prediction based on imputed data (fine mesh), and $\log \mathbb{E}_I(x) \geq -11$ (filled contour plot).

9.1.3.3 Step 3: Select Stopping Criteria

For noise corrupted problems, the most reliable stopping criterion is a limit on the number of function evaluations allowed. For this problem, an upper bound of $N = 50$ is used.

9.1.3.4 Step 4: Select Covariance Function

XFOIL uses the same models in all regions of the feasible design space to compute the sectional properties. The underlying physics are also expected to be smooth. Thus, there is no reason to believe that the function $c_d(\cdot)$ is nonsmooth. The surface shown in Figure 76 confirms this. The isotropic squared exponential covariance function is chosen for fBcEGO.

9.1.3.5 Step 5: Execute fBcEGO

The test matrix for this problem is shown in Table 16. fBcEGO is tested with three imputation methods from §5.3 and two initial designs. These methods are in turn compared with the GA and DIRECT, both of which have been modified to deal with hard constraints.

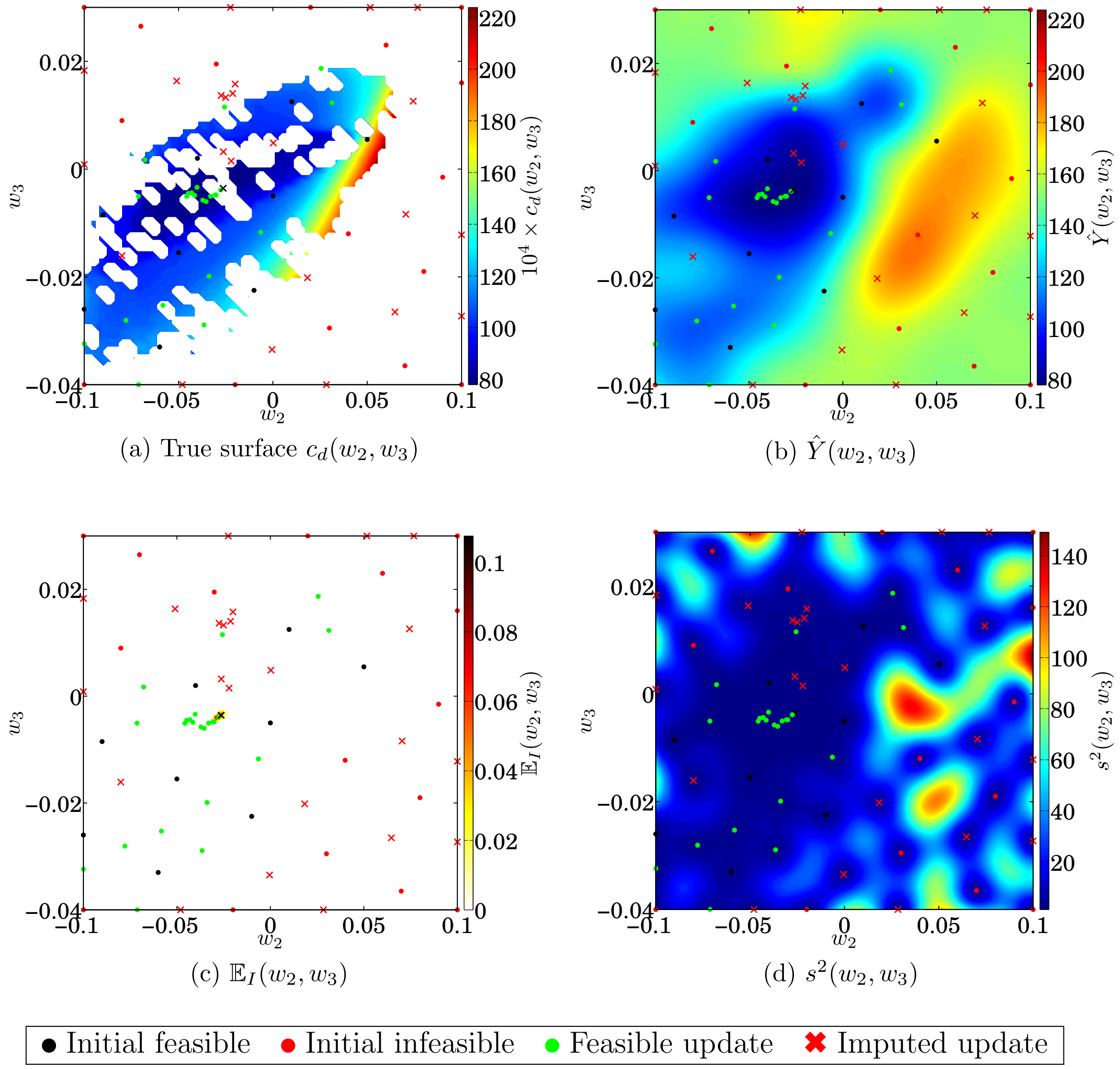
9.1.3.6 Step 6: Visualization & Analysis

Table 24 shows the results for the two-variable problem after $N = 50$ simplex gradients. All algorithms (except DIRECT) show similar failure rates in the initial design/population which reflects the failure rate of the surface. When fBcEGO employs the predictor imputation method, the subsequent failure rate approaches 90% for both designs. When the penalized imputation method is used, the subsequent failure rate is reduced by a factor of two. This demonstrates the ability of the penalized imputation method to drive the search away from the infeasible regions, which also results in lower values of c_d. The performance of fBcEGO with the penalized imputation method and N_2 LHD+FCP initial design after seventy one function evaluations where its best value of $10^4 \times c_d = 77.8$ is found is shown in Figure 77. Maximum value imputation provides better performance than predictor imputation because it penalizes infeasible points, but it is surpassed by the penalized imputation method. With the initial population of the GA concentrated within the feasible region, subsequent failures are low. Although the GA deals well with missing data, such a search is more suited to finding optimal regions in high-dimensional or multimodal problems rather than accurately locating the global minimum. `DIRECT` exhibits a low subsequent failure rate because the problem is unimodal and the algorithm samples frequently within the single global minimum basin, resulting in a search over a fine grid around the global minimum. The performance of `DIRECT` is compared with the performance of `fBcEGO` with N_2 LHD+FCP initial design and penalized imputation method after $N = 50$ simplex gradients in Figure 78. `fBcEGO` becomes trapped in the large flat drag bucket of the c_d surface and also continues to sample globally, while `DIRECT` samples a fine grid around the global minimum.

Figure 79 shows the optimized airfoil superposed with the original NASA SC(2)-0610 airfoil, and Figure 83a shows the surface pressure distribution of the optimized airfoil. It can be seen in Figure 79 that the camber in the aft section of the airfoil has increased. Comparing the surface pressure distribution of the optimized airfoil in Figure 83a with the distribution of the baseline airfoil in Figure 73b shows that the peak pressures at the upper surface of the leading edge and the lower surface of the trailing edge have been reduced.

9.1.4 Methodology Applied To Four-Variable Optimization

Table 25 shows the results for the four-variable problem. The performance trends are identical to those of the two-variable optimization, with `DIRECT` finding the best value of $10^4 \times c_d = 74.5$ followed by the penalized imputation method with N_2 LHD+FCP initial design with $10^4 \times c_d = 75.5$. The failure rates are higher because the infeasible



(a) True surface $c_d(w_2, w_3)$

(b) $\hat{Y}(w_2, w_3)$

(c) $\mathbb{E}_I(w_2, w_3)$

(d) $s^2(w_2, w_3)$

Figure 77: Performance of `fBcEGO` on the two-variable optimization problem after seventy one function evaluations with N_2 LHD+FCP initial design and penalized imputation method.

Table 24: Performance comparison of four methods for two-variable airfoil optimization after $N = 50$ simplex gradients.

Algorithm	Imputation	Initial design	Initial failures	Subsequent failures	$y_{\min}$
fBcEGO	Predictor	N_1 DGS+FCP	64.71%	87.22%	78.7
fBcEGO	Predictor	N_2 LHD+FCP	64.00%	85.60%	80.6
fBcEGO	Penalized	N_1 DGS+FCP	64.71%	39.10%	78.3
fBcEGO	Penalized	N_2 LHD+FCP	64.00%	47.20%	77.8
fBcEGO	Max. value	N_1 DGS+FCP	64.71%	57.14%	77.6
fBcEGO	Max. value	N_2 LHD+FCP	64.00%	59.20%	79.3
GA[a]	Discard	Random	70.66%	27.21%	78.9[b]
DIRECT	Max. value	–	–	24.67%	77.6

[a] Average of 10 runs with initial population of 10

[b] Standard deviation of ±1.8; min. value found was $10^4 \times c_d = 77.5$

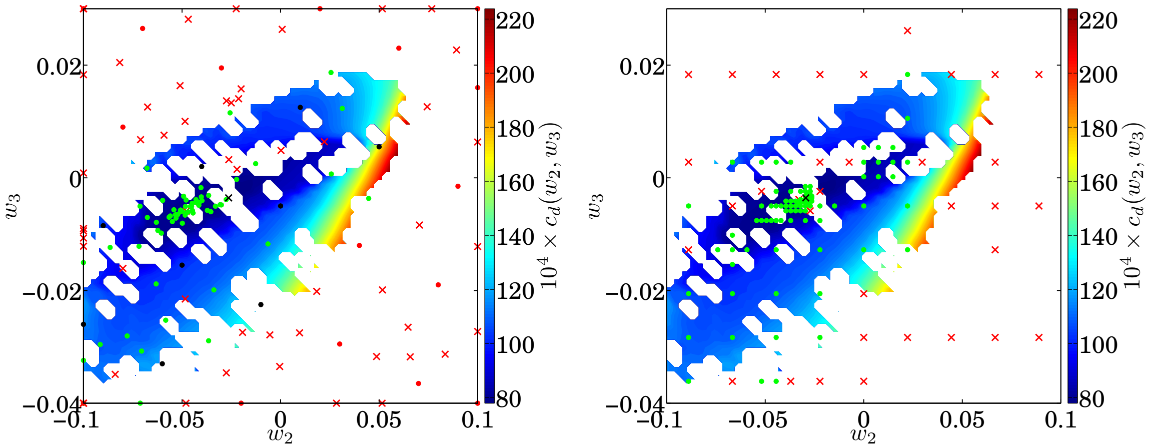


Figure 78: Contour plot of $10^4 \times c_d(w_2, w_3)$ with performance of `fBcEGO` (left) and `DIRECT` after $N = 50$ simplex gradients (right); see Figure 77 for legend.

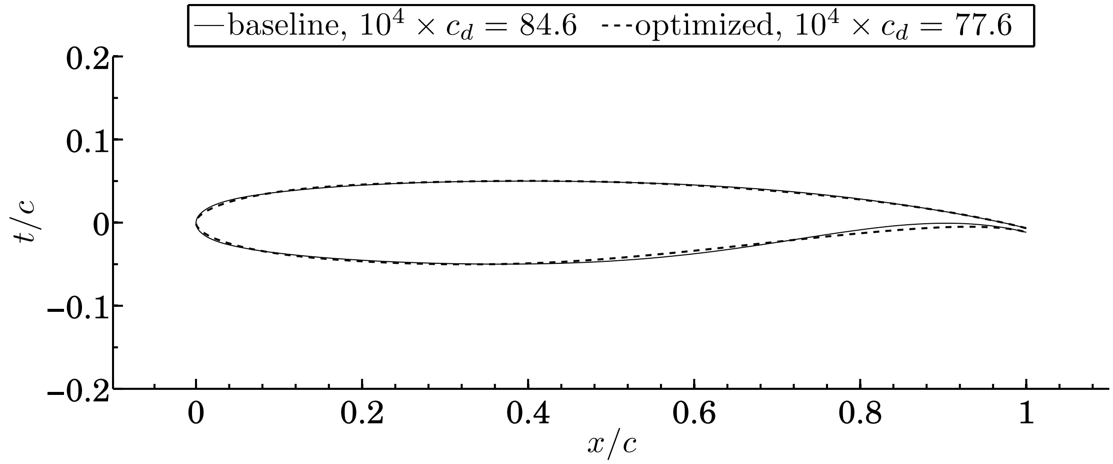


Figure 79: Baseline versus optimized geometry for two-variable optimization of the SC(2)-0610 airfoil.

region is larger. Figure 80 shows the performance of `fBcEGO` with N_2 LHD+FCP initial design and with the penalized imputation method after $N = 50$ simplex gradients. Figure 81 shows the performance of `DIRECT` after $N = 50$ simplex gradients. These two figures show a grid of two-dimensional tiles, each with w_2 and w_3 varying over their range but with a discrete value of w_4 and w_5. As such, the design variables for w_4 and w_5 were rounded to the nearest multiple of 0.03 for visualization purposes. The true surface $c_d(w_2, w_3, w_4, w_5)$ is built from a simulation of 11^4 XFOIL runs and is represented by the colored patches in the figure. The failure rate of this "true" surface was 91.64%. The dark colored patches represent areas that are infeasible due to nonsensical airfoil geometry. Figure 80 and Figure 81 shows a large number feasible points clustered around the global minimum. Including the additional basis function f_4 and f_5 in the airfoil parameterization has increased the geometry control at the leading edge, allowing more efficient airfoils to be generated. The drag is reduced by thirty one counts (4.0%) from the two-variable global minimum when all four basis functions are used. Figure 82 shows the optimized airfoil superposed with

Table 25: Performance comparison of four methods for four-variable airfoil optimization after $N = 50$ simplex gradients.

Algorithm	Imputation	Initial design	Initial failures	Subsequent failures	$y_{\min}$
fBcEGO	Predictor	N_1 DGS+FCP	71.15%	98.99%	77.8
fBcEGO	Predictor	N_2 LHD+FCP	94.12%	75.88%	77.8
fBcEGO	Penalized	N_1 DGS+FCP	71.15%	96.46%	77.5
fBcEGO	Penalized	N_2 LHD+FCP	94.12%	74.87%	75.5
fBcEGO	Max. value	N_1 DGS+FCP	71.15%	87.37%	77.5
fBcEGO	Max. value	N_2 LHD+FCP	94.12%	75.87%	78.2
GA[a]	Discard	Random	89.82%	47.75%	84.2[b]
DIRECT	Max. value	–	–	37.20%	74.5

[a] Average of 10 runs with initial population of 20

[b] Standard deviation of ±6.4; min. value found was $10^4 \times c_d = 78.3$

the original NASA SC(2)-0610 airfoil, and Figure 83b shows the surface pressure distribution of the four-variable optimized airfoil. The improved geometry control of the four-variable problem at the leading edge results in a smoother leading edge pressure distribution. Compared with the original surface pressure distribution (Figure 73a), the optimized airfoil exhibits lower peak pressure coefficients.

9.1.5 Conclusions

The use of imputation methods to handle hard constraints enables optimization of unfamiliar design spaces. The penalized imputation method outperformed the predictor and maximum value imputation methods in terms of subsequent failure rate on the two-variable problem. The performance of the imputation methods on the four-variable problem did not exhibit any discernible differences. This was due to the difficulty of the design space.

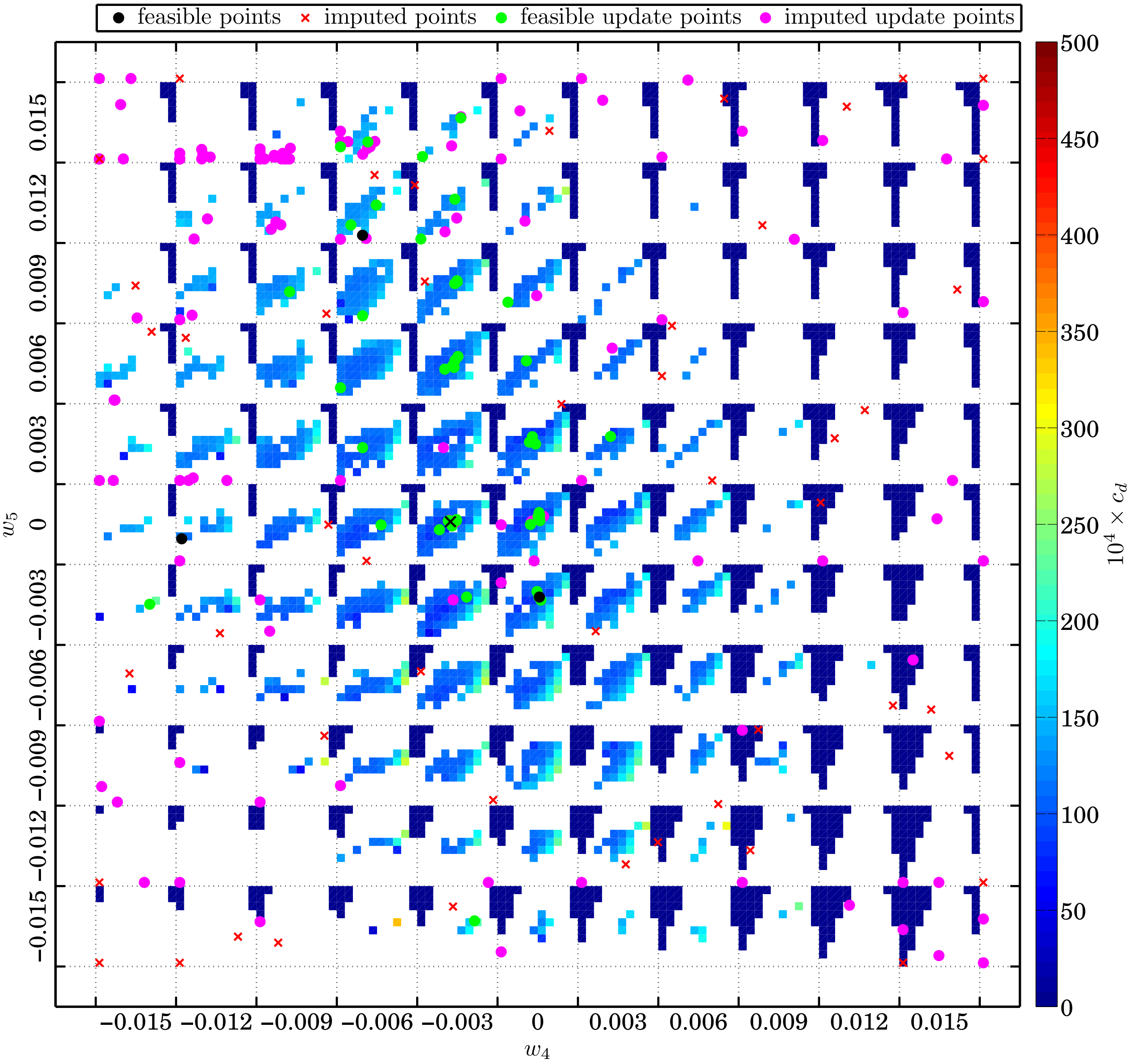


Figure 80: Performance of fBcEGO on the four-variable optimization problem after $N = 50$; N_2 LHD+FCP initial design with penalized imputation method. Regions of infeasibility due to nonsensical airfoil geometries are shown as dark blue.

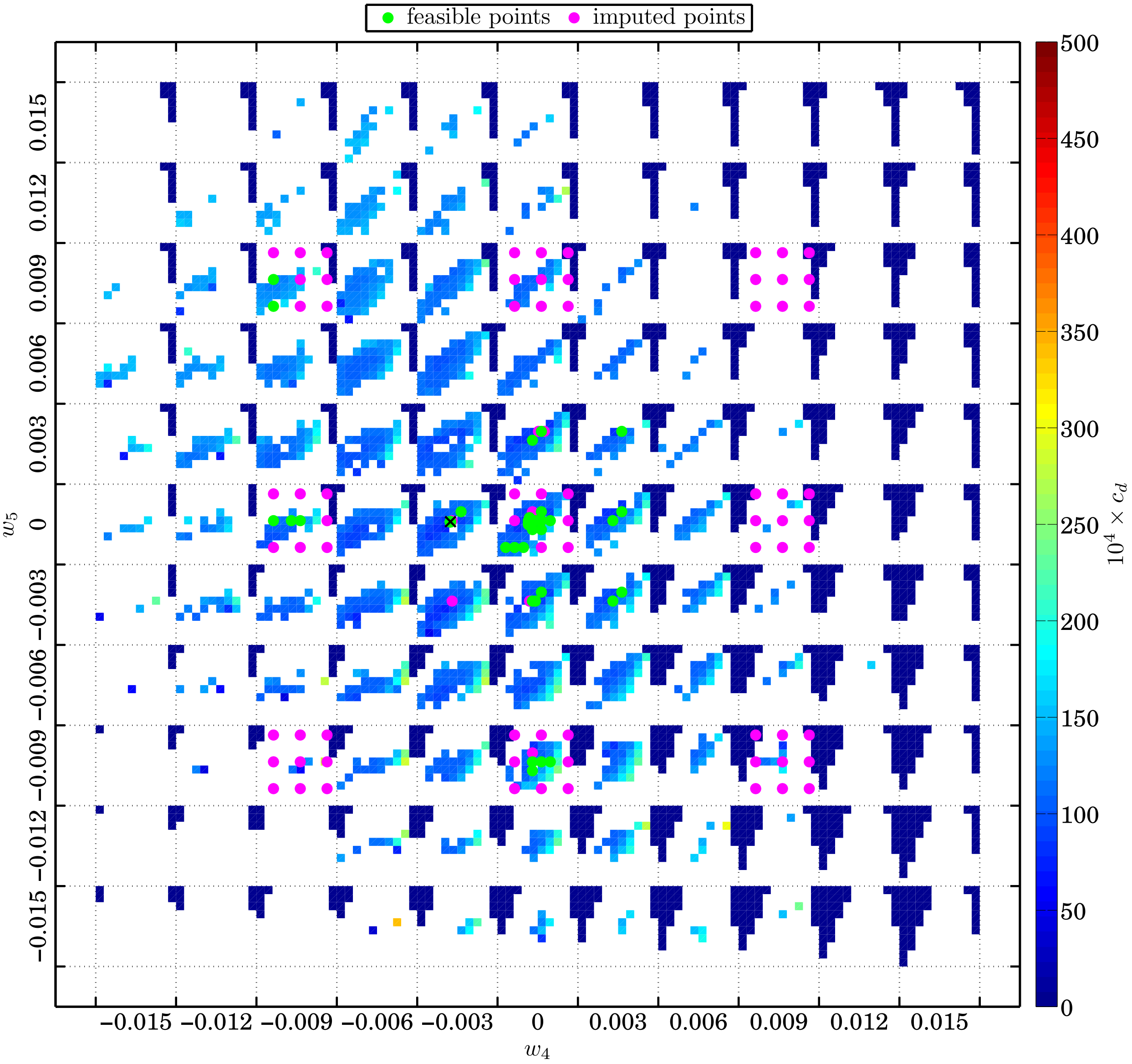


Figure 81: Performance of DIRECT on the four-variable optimization problem after $N = 50$ with maximum value imputation method. Regions of infeasibility due to nonsensical airfoil geometries are shown as dark blue.

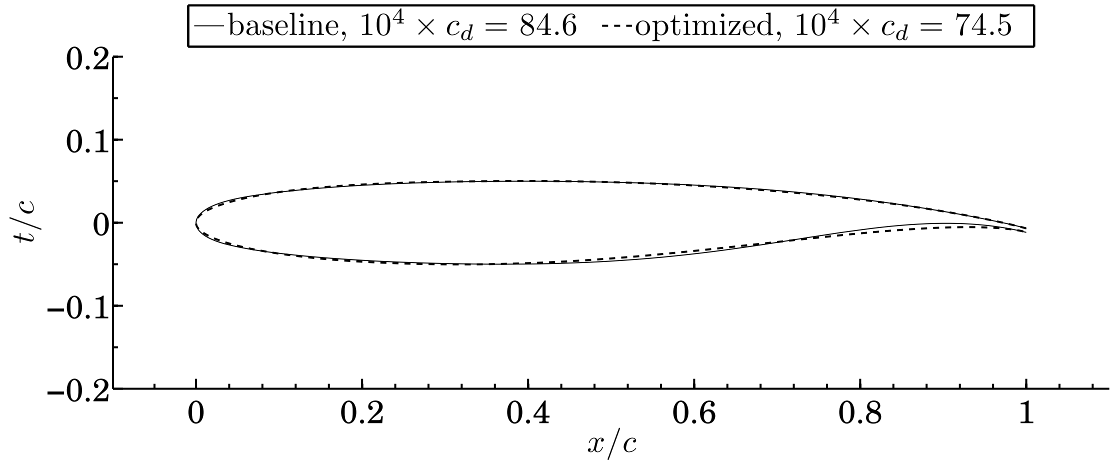
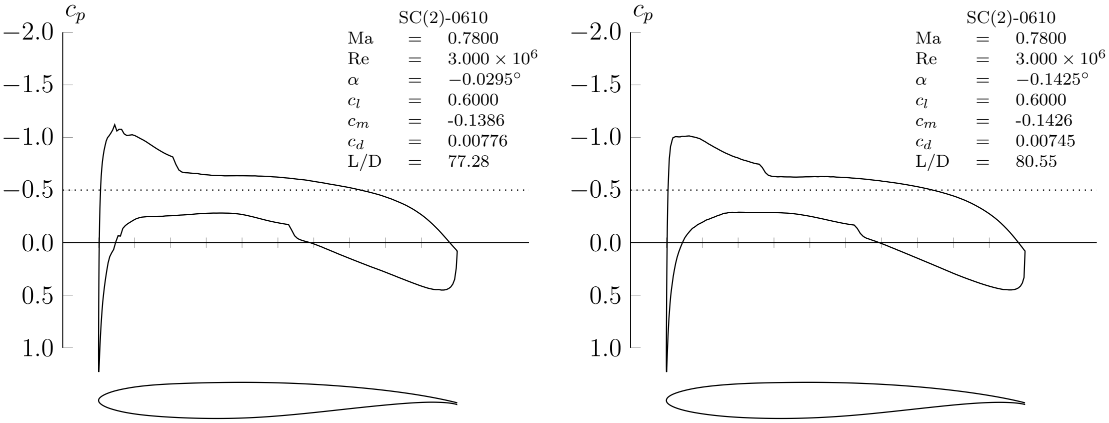


Figure 82: Baseline versus optimized geometry for four-variable optimization of SC(2)-0610 airfoil.



(a) Two-variable problem

(b) Four-variable problem

Figure 83: Surface pressure distribution of optimized airfoils.

Figure 84: Bombardier CRJ-700 [www.aviationnews.eu].

9.2 Design Of A Notional 70-Passenger Aircraft

9.2.1 Background & Challenges

The purpose of this application is to test the performance of `fBcEGO` on a larger design problem that is subject to all the technical challenges listed in §1.2. The aircraft of interest is a notional 70-passenger regional jet modeled after the Bombardier CRJ-700 (see Figure 84). During the period between 1995 and 2004, regional jet fleets in the United States have increased exponentially. Between 1998 and 2003, daily regional jet operations increased by 356% [72]. The demand for regional aircraft in the 60- to 99-seat segment is expected to continue to grow over the next twenty years [16].

The design philosophy for this aircraft design problem is one of determining the combination of geometric parameters that will meet performance requirements at minimum weight. The weight is considered as the traditional indicator of the overall life-cycle cost of the aircraft. Analysis is provided by the Flight Optimization System (FLOPS) [65], a multidisciplinary design code for conceptual aircraft design (see §9.2.2). The mission profile of the aircraft is given in Figure 85.

The design space of this problem is subject to all the technical challenges listed in §1.2 except for the computational expense. Figure 86 shows some responses from FLOPS as functions of thrust-to-weight ratio and wing area in the region surrounding

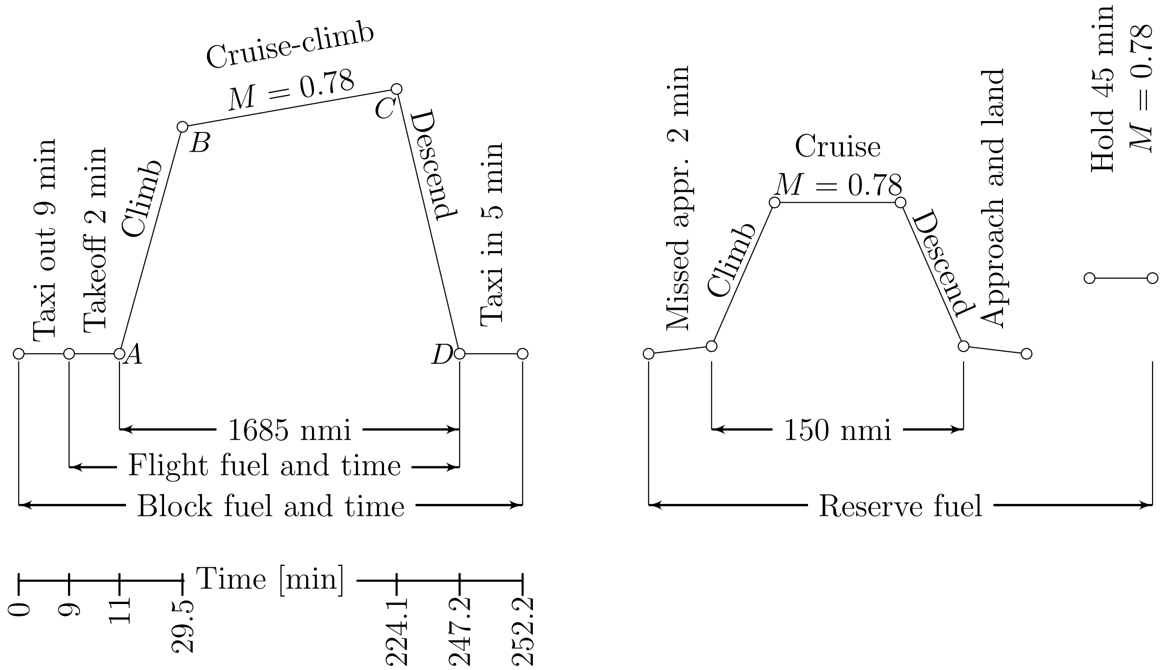
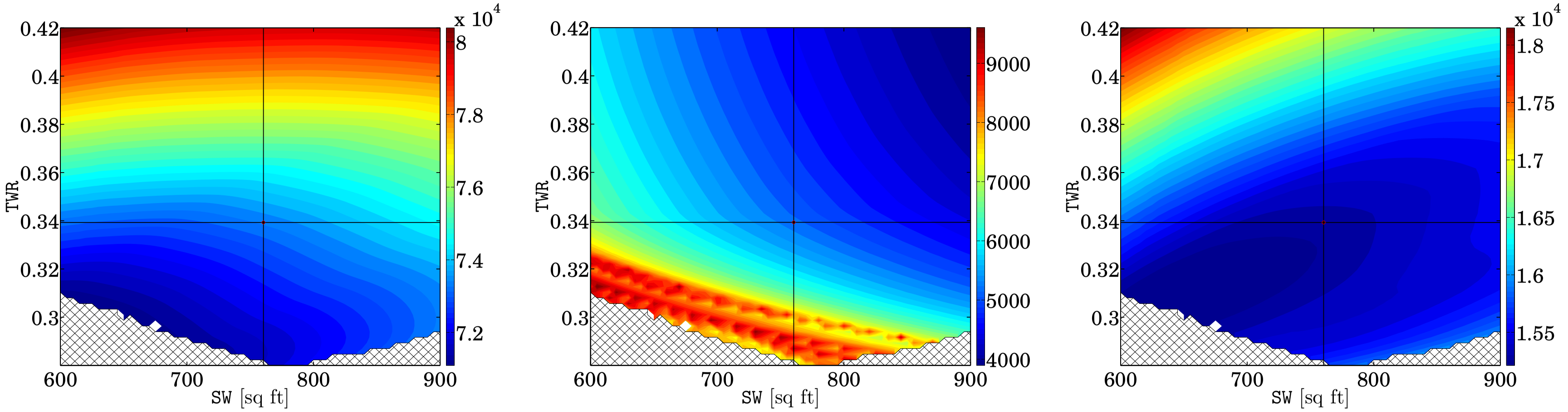


Figure 85: Mission profile for notional 70-passenger jet, shown in the order in which it is calculated by FLOPS. Left, main mission; right, reserve mission. Altitude and distance traveled at specified points: 0 ft, 0 nmi at A; 34266 ft, 121.5 nmi at B; 37593 ft, 1455 nmi at C; 0 ft, 1685 nmi at D.

the baseline aircraft. The design space is generally nonconvex with regions which are infeasible due to hard constraints. Some responses are poorly behaved on the boundaries of the feasible region and appear noisy. In addition, the design variable bounds are generally unknown but assumed given for this problem.



(a) Gross weight `TOGW` [lb]

(b) Landing field length `LDGFL` [ft]

(c) Block fuel `FW` [lb]

Figure 86: Some responses as functions of wing area `SW` [sq ft] and thrust-to-weight ratio `TWR`, with the remaining variables set to their baseline values. Baseline indicated by crosshairs. Regions that are infeasible due to hard constraints are indicated by hatched areas.

9.2.2 Flight Optimization System (FLOPS) [65]

The Flight Optimization System (FLOPS) is a multidisciplinary system of computer programs for conceptual design and analysis of advanced aircraft. FLOPS is generally used for subsonic commercial aircraft but has been extended to supersonic commercial aircraft [60]; for military systems, the Aircraft Synthesis tool (ACSYNT) [80] is more appropriate. The ability to integrate customized models for the different aircraft design disciplines into the program makes FLOPS suitable for the conceptual design and analysis of advanced aircraft concepts. FLOPS consists of nine modules: weights, aerodynamics, engine cycle analysis, propulsion data scaling and interpolation, mission performance, takeoff and landing, noise footprint, cost analysis, and program control. Through the program control module, FLOPS may be used to analyze a point design, parametrically vary design variables, or optimize a configuration for performance using internal nonlinear programming techniques. In this application, FLOPS is treated as a black-box analysis tool and `fBcEGO` is used as the optimizer.

The weights module uses empirical equations to predict the weight of each item in a group weight statement. Analytical wing weight estimation techniques are available for some unconventional wing planforms. The aerodynamics module is based on an empirical drag estimation technique developed from an analysis of nineteen subsonic and supersonic military aircraft and fifteen supercritical airfoil configurations [33] and has been modified to include a method for skin friction calculations at high Mach numbers [111]. Alternatively, drag polars may be input and then scaled with variations in wing area and engine nacelle size. The engine cycle analysis module is based on [43] and provides the capability to internally generate an engine deck consisting of thrust and fuel flow data at a variety of Mach-altitude conditions. The propulsion data scaling and interpolation module uses an engine deck that has been input or one that has been generated by the engine cycle analysis module and uses linear or nonlinear scaling laws to scale the engine data to the desired thrust. The mission performance module uses the weights, aerodynamics, and propulsion data to calculate flight performance for a given mission profile. The mission segments may be flown at a variety of conditions, e.g., cruise at optimum altitude or Mach number or for maximum range or endurance. The takeoff and landing module computes field lengths and performance in accordance with FAA Federal Aviation Regulations Part 25 [1]. A detailed takeoff and climbout profile can also be generated for use in calculating noise footprints. The noise footprint module [24] generates contour data of the noise levels at user-specified or FAA locations at takeoff and climbout. Noise sources include fan inlet and exhaust, jet, flap, combustor, turbine, and airframe. Noise propagation corrections are available for atmospheric and ground attenuation, ground reflections, and shielding. The cost module in this version of FLOPS (6.12) was replaced with the more detailed cost module called Aircraft Life Cycle Cost Analysis (ALCCA) [64], but a cost analysis was not performed.

9.2.3 Competing Methods

The primary algorithm for this problem is `fBcEGO`. The penalized imputation method is utilized and the `fBcEGO` is tested with two initial designs: N_1 DGS+FCP and N_2 LHD+FCP. Two additional competing algorithms are considered for this problem: `DIRECT` with the maximum value imputation scheme and a genetic algorithm. The implementation and associated details of the algorithms are identical to that of the previous design problem (see §9.1.2).

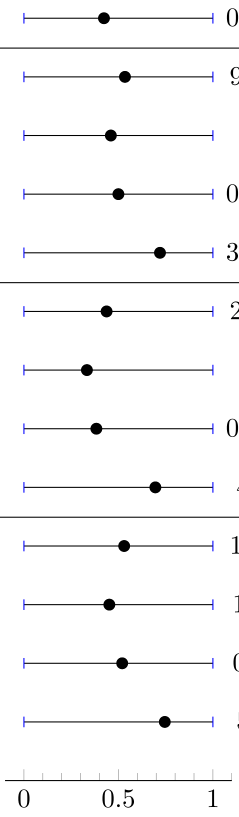
9.2.4 Methodology Applied To Aircraft Design Problem

9.2.4.1 Step 1: Formal Problem Statement

There are thirteen design variables in total: twelve geometric variables dealing with the configuration of the main wing and the tail surfaces, and one variable for the thrust-to-weight ratio. The design variables, their baseline values, and their bounds

Table 26: Aircraft design variables and ranges with relative normalized position of baseline values shown graphically

	Design Variable	Nomenclature	Baseline	x^{ℓ}		x^{u}
	Thrust/weight	`TWR`	0.339296	0.28		0.42
Main wing	Area [ft^2]	`SW`	760.4	600		900
	Aspect ratio	`AR`	7.38	6		9
	Taper ratio	`TR`	0.32	0.28		0.36
	Sweep [deg]	`SWEEP`	27	0		37.5
Horizontal tail	Area [ft^2]	`SHT`	193.74	150		250
	Aspect ratio	`ARHT`	4	3		6
	Taper ratio	`TRHT`	0.465	0.35		0.65
	Sweep [deg]	`SWPHT`	31.3	0		45
Vertical tail	Area [ft^2]	`SVT`	133.04	80		180
	Aspect ratio	`ARVT`	1.089	0.75		1.5
	Taper ratio	`TRVT`	0.656	0.5		0.8
	Sweep [deg]	`SWPVT`	41	0		55

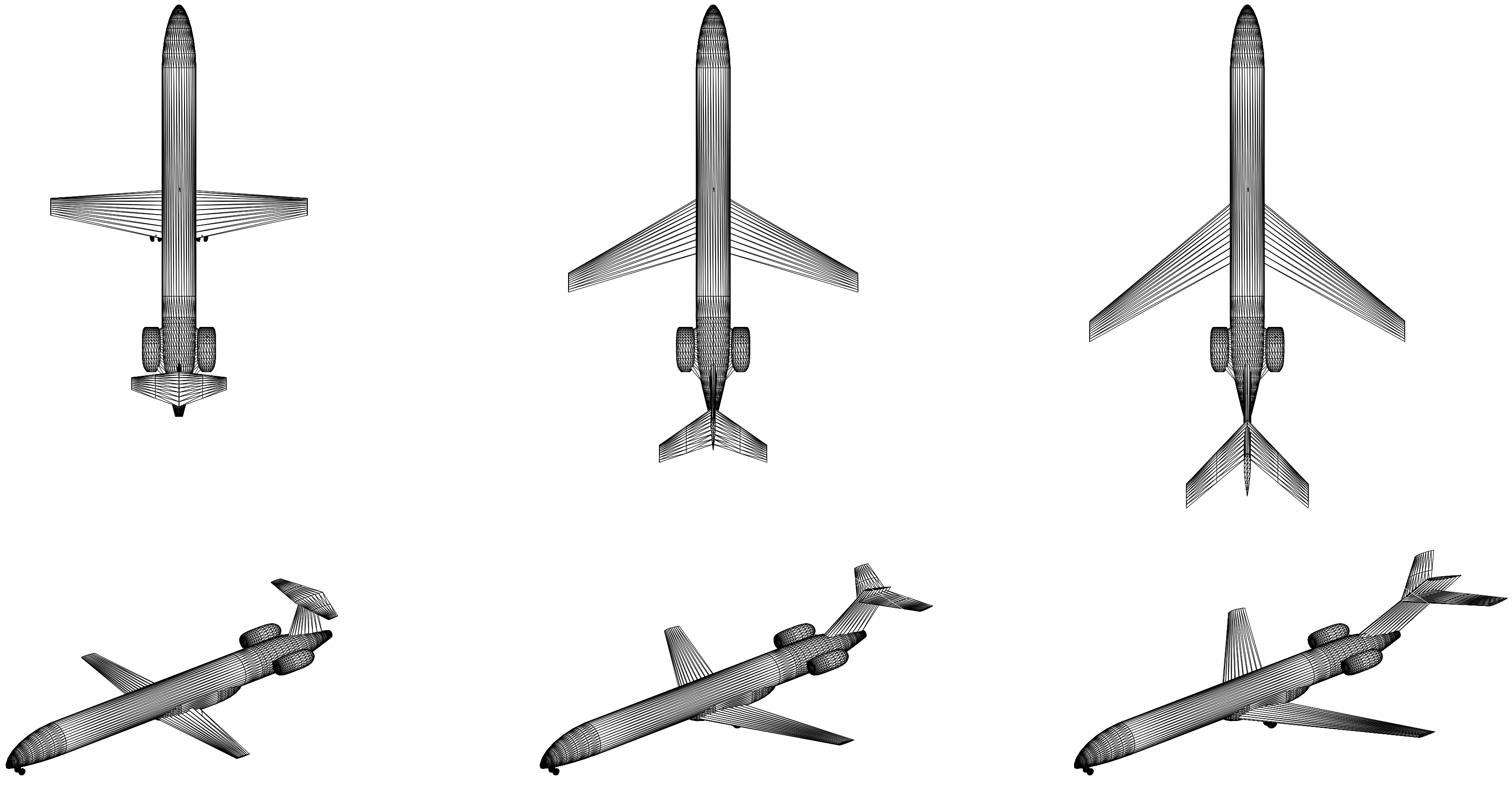


are given in Table 26. The aircraft configurations corresponding to the lower and upper bounds are visualized in Figure 87 against the baseline. The bounds are generally unknown and assumed to be given, but as with the airfoil design problem, the design space can be bounded by inspecting the aircraft configurations at the bounds.

There are five nonlinear constraints relating to the aircraft performance, which are given in Table 27 along with their baseline values and constraint limits. The takeoff gross weight `TOGW` of the aircraft is to be minimized while simultaneously meeting or exceeding the performance of the baseline aircraft. A formal mathematical statement of this optimization problem is giving in problem 9.1.

Table 27: Aircraft design objective and constraints.

Response	Nomenclature	Baseline Values	Constraint Bound	Constraint Δ	Units
Takeoff gross weight	`TOGW`	73465.3	minimize	–	lb
Takeoff field length	`TOFL`	5233	$\leq$ 5233	0.00%	ft
Landing field length	`LDGFL`	5019	$\leq$ 5019	0.00%	ft
Approach speed	`VAPP`	133.4	$\leq$ 133.4	0.00%	kts
Oper. empty weight	`OEW`	43903.7	$\leq$ 43903.7	0.00%	lb
Fuel weight	`FW`	15321.5	$\leq$ 15321.5	0.00%	lb



(a) Lower bounds

(b) Baseline

(c) Upper bounds

Figure 87: Visualization of design space bounds by aircraft configuration.

$$
\begin{aligned}
\underset{x \in \mathbb{R}^{13}}{\text{minimize}} \quad & \texttt{TOGW} \\
\text{subject to} \quad & 1 - \frac{\texttt{TOFL}}{5233} \geq 0 \\
& 1 - \frac{\texttt{LDGFL}}{5019} \geq 0 \\
& 1 - \frac{\texttt{VAPP}}{133.4} \geq 0 \\
& 1 - \frac{\texttt{OEW}}{43903.7} \geq 0 \\
& 1 - \frac{\texttt{FW}}{15321.5} \geq 0 \\
& x^{\ell} \leq x \leq x^{u}
\end{aligned}
\tag{9.1}
$$

9.2.4.2 Step 2: Select Initial Design

Two initial designs are chosen for `fBcEGO`: N_1 DGS+FCP and N_2 LHD+FCP. The implementation of these designs is identical to that of the previous design problem.

9.2.4.3 Step 3: Select Stopping Criteria

The global minimum value for this problem is unknown. The global minimizer may not even occur within the chosen bounds for this problem. In addition, it is not known if the current baseline is already the global minimum, i.e., it may not be possible to further reduce the weight of the aircraft without violating one or more constraints. The stopping criterion for this problem is set by an upper limit on the number of simplex gradients, $N = 20$ in this case.

9.2.4.4 Step 4: Select Covariance Function

Based on Figure 86, the design space appears to be well-behaved on the interior. However, because this is a black-box design problem, the properties of the responses may never truly be known. For this reason, the isotropic squared exponential covariance function is chosen for `fBcEGO`.

9.2.4.5 Step 5: Execute `fBcEGO`

`fBcEGO` is tested with the penalized imputation method and two initial designs. These methods are in turn compared with the GA and `DIRECT`, both of which have been modified to deal with hard constraints. All algorithms terminate after $N = 20$ simplex gradients have been evaluated, i.e., after 280 function evaluations have been made (initial design inclusive).

9.2.4.6 Step 6: Visualization & Analysis

The results of the aircraft design problem are presented in this section. The performance of `fBcEGO` with N_2 DGS+FCP initial design is discussed first, since this design resulted in a lighter aircraft versus the N_1 LHD+FCP initial design. The optimized configuration (wireframe) is shown in Figure 88 superposed with the baseline configuration (solid model). Tabulated results are shown in Table 28. With respect to the objective function, `fBcEGO` reduced the takeoff gross weight by 9.87% from the baseline value. The sensitivity chart (Figure 89) shows that this is primarily a result of the low thrust-to-weight ratio and the small tail surface areas.

The landing field length remains constant and is an active constraint. The decrease in the thrust-to-weight ratio was offset by a corresponding increase in the wing area, retaining the level of this response. All variables except the thrust-to-weight ratio have been selected at the limits of the design space, i.e., the bound constraints are active for these variables. This result is a consequence of not knowing the bound constraints to the black-box problem and indicates that the design space should be expanded to allow `fBcEGO` to search for better designs. This situation is accounted for in the methodology in Figure 43.

Contour profiles for the baseline and the optimized designs are shown in Figure 90. These plots were generated by evaluating FLOPS over a grid of values for the thrust-to-weight ratio and the wing area and setting the remaining variables to either the baseline or optimized values, respectively. The contour profiler at the optimum shows no regions that are infeasible by hard constraints with respect to thrust-to-weight ratio and the wing area. In addition, at the optimum, the feasible area with respect to thrust-to-weight ratio and the wing area is too small to be seen. A difficulty that may be encountered in the optimization of black-box design problems is the lack of a feasible region. fBcEGO addresses this by employing a "Phase I"-type approach if no feasible points exist in the initial design (see §5.2). If no feasible values have been found when fBcEGO terminates, it is likely that a feasible region does not exist. In this situation, the constraints can be relaxed commensurate with the customer's updated requirements.

The performance of fBcEGO is compared with the competing methods in Table 29. In general, each method found a different solution when compared with the remaining methods, indicating the possibility that the design space is multimodal. These configurations are visualized in Figure 91. With respect to the takeoff gross weight, all algorithms improved from the baseline but fBcEGO returned the lightest aircraft, which was over 1000 pounds lighter than the next lightest aircraft found by DIRECT. With the exception of the takeoff field length, no major differences can be seen in the performance of each optimized design. The GA returns an aircraft configuration which reduces the takeoff field length by 12.77%. The landing field length and approach speed were also correspondingly lower than the other configurations, indicating the possibility of a large feasible area at the configuration found by the GA.

Table 30 shows the percentage of failed and infeasible designs in both the initial design/population and subsequent iterates. The failure rate of the N_2 LHD+FCP design was 65.4% and the failure rate of the initial population of the GA was 57.1%. These failure rates reflect the failure rate of the design space due to hard constraints, which was approximately 60%. The N_1 DGS+FCP design showed a significantly lower failure rate due to the adaptive nature of DIRECT. DIRECT shows no initial failure rate because DIRECT does not use an initial design. The subsequent failure rate, which is the failure rate of the iterates beyond the initial design, was 0.8% (one failed iterate) for fBcEGO with N_2 LHD+FCP initial design and 4.8% (7 failed iterates) for fBcEGO with N_1 DGS+FCP initial design. fBcEGO uses the penalized imputation method to address hard constraints and the results indicate the effectiveness of this strategy in driving iterates away from failed regions.

The percentage of infeasible designs in the initial design/population and subsequent iterates is indicated in the last column of Table 30. A design can be infeasible either by violating an inequality constraint or a hard constraint, thus, this column includes failed iterates as infeasible designs. The percentage of infeasible designs in the initial designs of fBcEGO and the GA reflects the size of the infeasible region of the design space, which was approximately 96.2%. The percentage of subsequent iterates that were infeasible was 26.5% for fBcEGO with N_1 DGS+FCP design and 57.9% for fBcEGO with N_2 LHD+FCP design. This indicates the importance of using adaptive initial designs. The GA found no feasible subsequent iterates and DIRECT also had a high percentage of infeasible iterates (88.9%) because it is limited to sampling along the coordinate directions.

Figures 92 – 95 show the iterates of the four methods in Table 30 in the wing area by thrust-to-weight ratio space. It can be seen from Figure 93 that fBcEGO with N_2 LHD+FCP initial design explores the design space more than fbcEGO with N_1 DGS+FCP initial design (Figure 92). Figure 94 (right) shows that only one sample from the GA was feasible. Figures 92, 93, and 95 show the feasible iterates converging towards designs with high wing area and low thrust-to-weight ratio (bottom right of the charts).

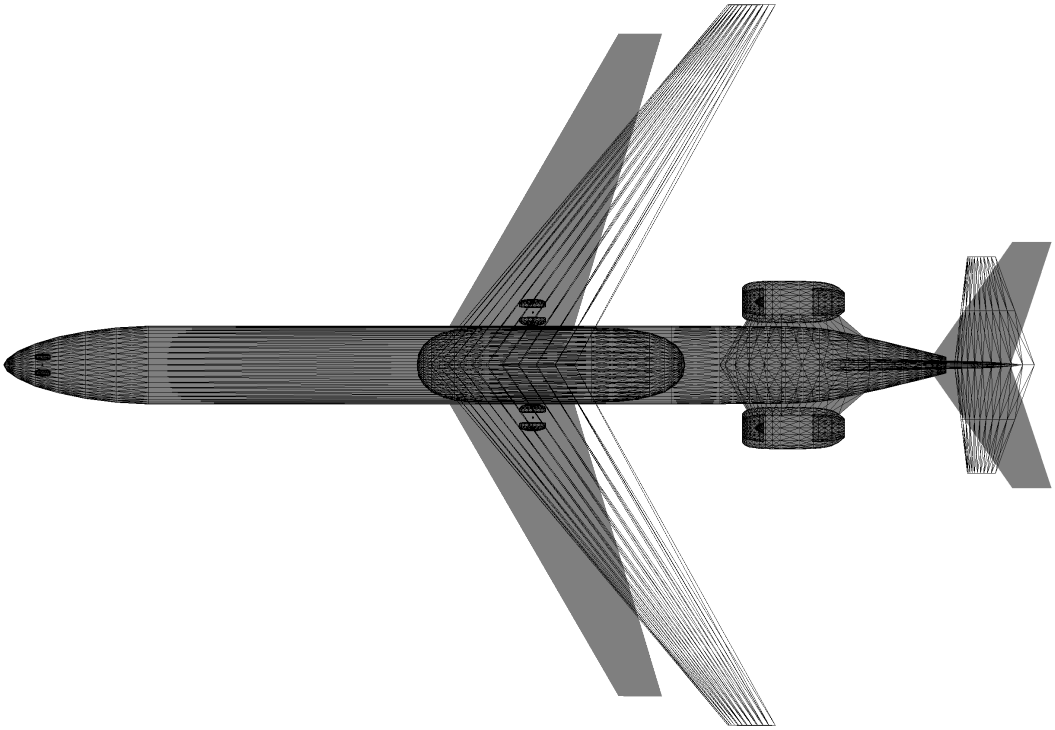
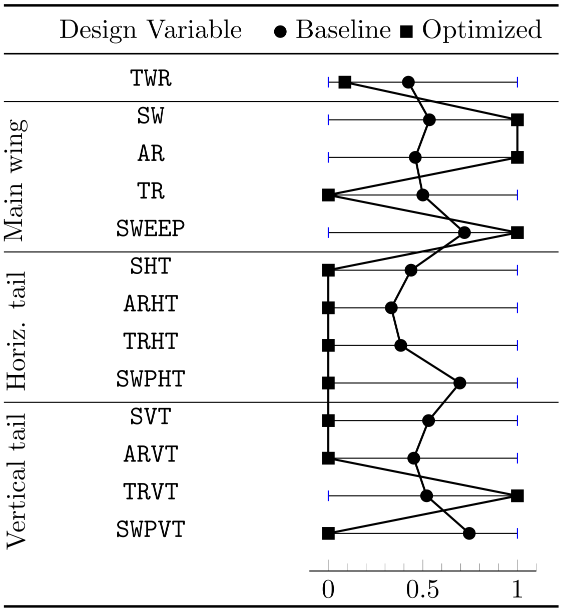


Figure 88: Baseline (solid) versus optimized (wireframe) configuration.

9.2.5 Conclusions

This design problem demonstrated the superiority of `fBcEGO` over two other methods when the function evaluation budget is severely limited. `fBcEGO` can provide significant time savings over other direct methods, such as `DIRECT` or GAs. This problem also demonstrates the importance of selecting appropriate bounds for the design space. The best solution was found at the bounds of the design space and a user may wish to expand one or more variables to find better solutions. This does not pose any difficulty for `fBcEGO`, since all previous function evaluations can be reused.

Table 28: Baseline versus optimized configuration found by fBcEGO. Design variables (left) and responses (right).



Response	▬ Baseline	▬ Optimized	Δ
TOGW	73,465.3 lb	66,211.8 lb	−9.87%
TOFL	5,233.0 ft	5,233.0 ft	0.00%
LDGFL	5,019.0 ft	4,383.0 ft	−12.67%
VAPP	133.4 kts	116.4 kts	−12.74%
OEW	43,903.7 lb	41,145.8 lb	−6.28%
FW	15,321.5 lb	11,679.0 lb	−23.77%

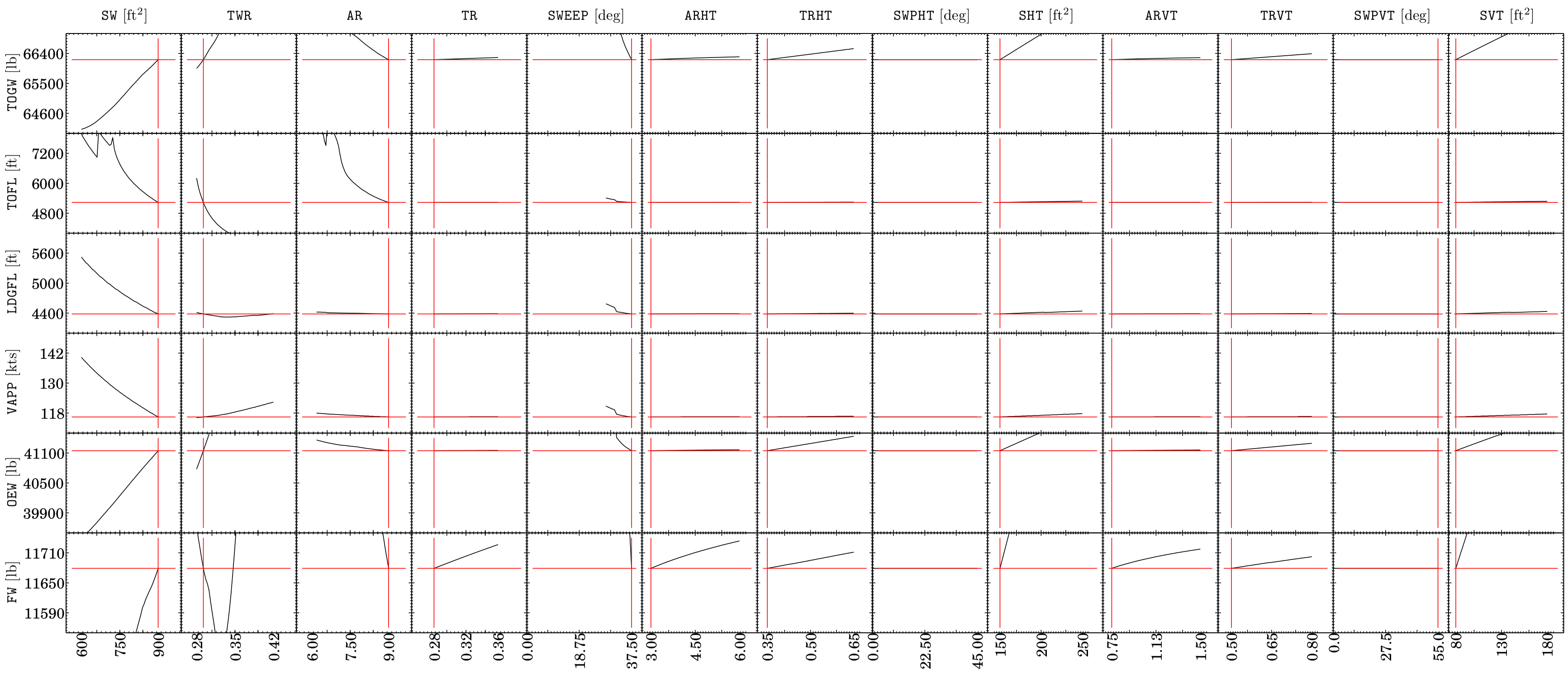


Figure 89: Sensitivity of responses to variables at the optimum; red crosshairs indicate optimum values.

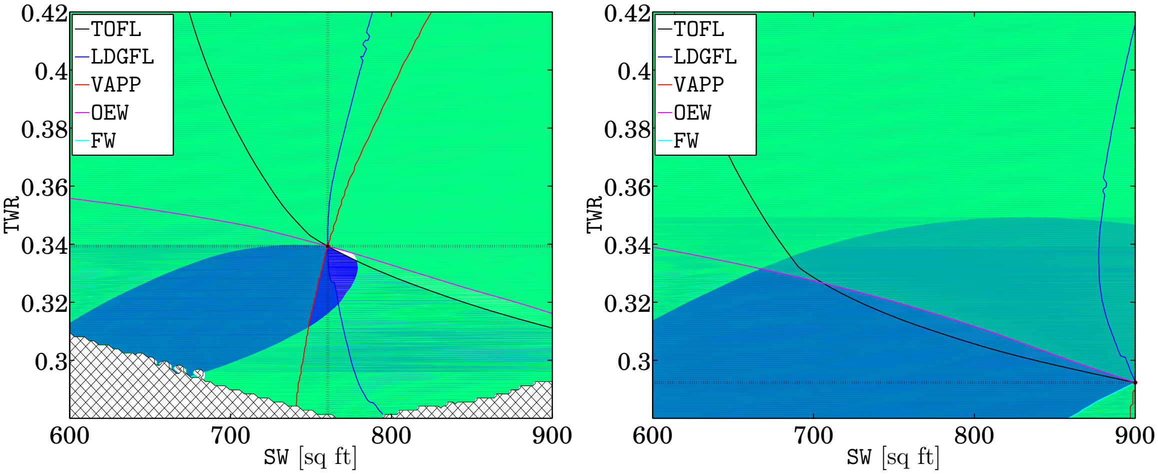
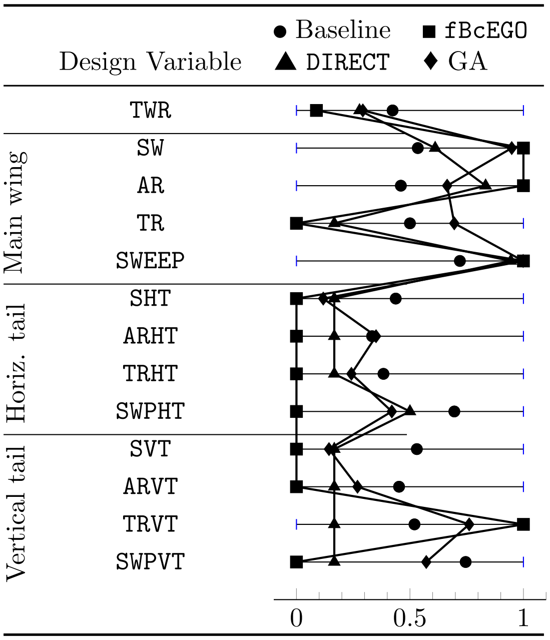
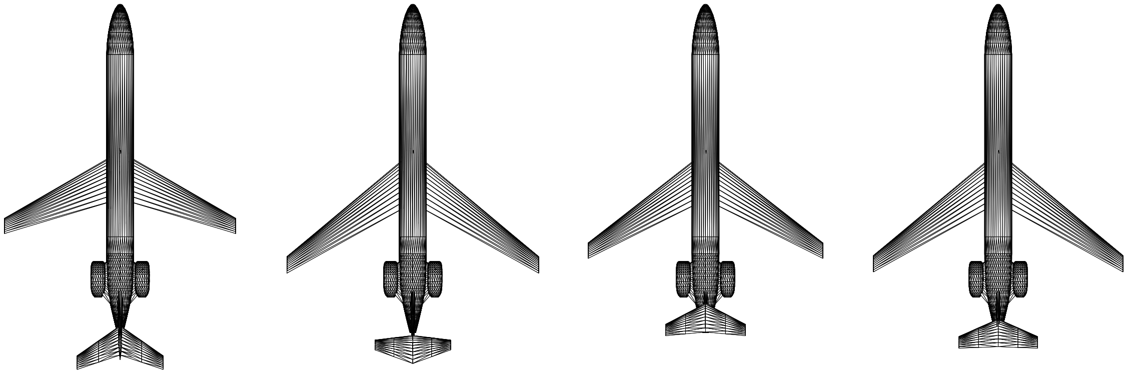


Figure 90: Contour profiles of the design space as a function of wing area SW [sq ft] and thrust-to-weight ratio TWR, with the remaining variables set to their corresponding values; baseline aircaft (left) and optimized aircraft (right). Designs indicated by crosshairs.

Table 29: Comparison of optimized configurations generated by competing methods, tabulated results. Design variables (left) and responses (right).



Response	■ Baseline	■ fBcEGO	■ DIRECT	□ GA
TOGW	73,465.3 lb	66,211.8 lb	67,260.9 lb	68,139.3 lb
TOFL	5,233.0 ft	5,233.0 ft	5,059.0 ft	4,565.0 ft
LDGFL	5,019.0 ft	4,383.0 ft	4,713.0 ft	4,410.0 ft
VAPP	133.4 kts	116.4 kts	125.8 kts	119.1 kts
OEW	43,903.7 lb	41,145.8 lb	41,723.4 lb	42,448.3 lb
FW	15,321.5 lb	11,679.0 lb	12,015.3 lb	12,121.8 lb



(a) Baseline (b) fBcEGO (c) DIRECT (d) GA

Figure 91: Comparison of optimized configurations generated by competing methods.

Table 30: Comparison of failed iterations and infeasible designs for each method.

Algorithm	■ Init. failures □ Sub. failures	■ Init. infeasible □ Sub. infeasible
fBcEGO N_1 DGS+FCP	45.11% 4.76%	98.50% 26.53%
fBcEGO N_2 LHD+FCP	65.41% 0.83%	96.23% 57.85%
GA	57.14% 17.14%	99.43% 100.00%
DIRECT	0.00% 15.71%	0.00% 88.93%

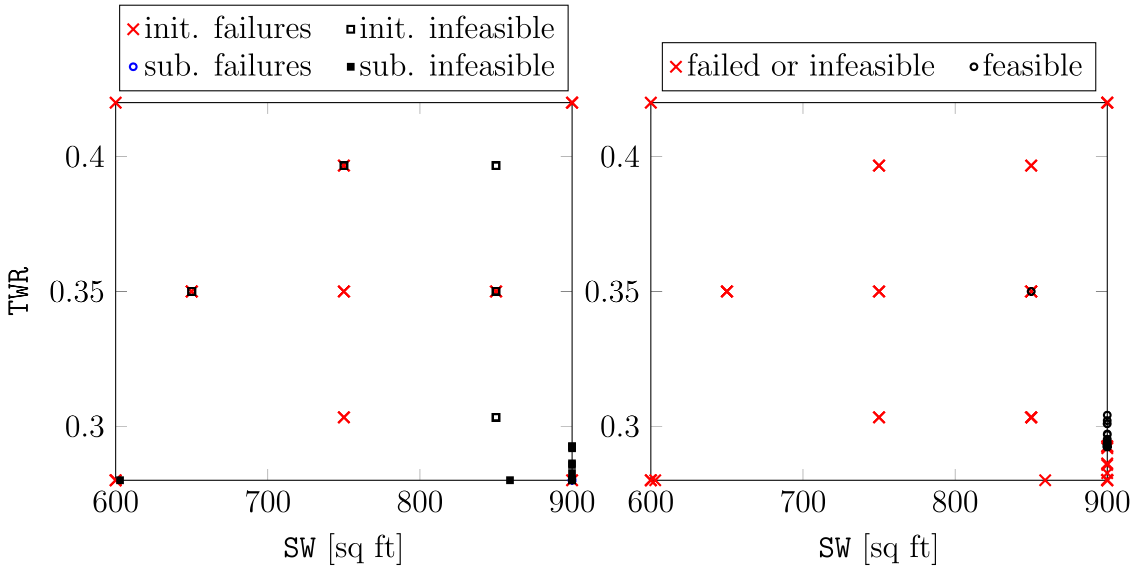


Figure 92: Sampling of fBcEGO with N_1 DGS+FCP initial design.

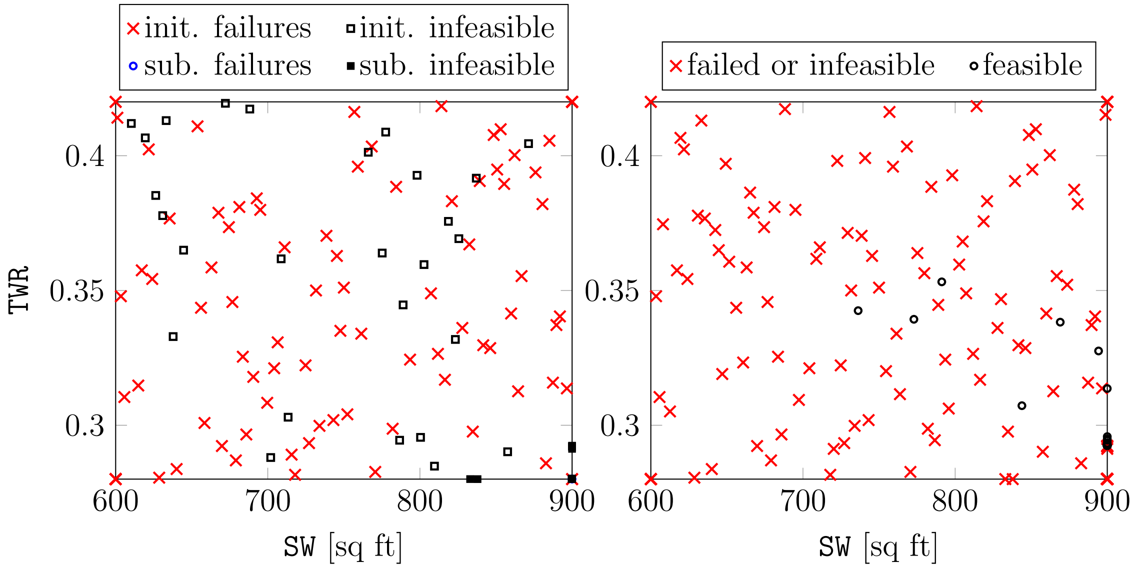


Figure 93: Sampling of fBcEGO with N_2 LHD+FCP initial design.

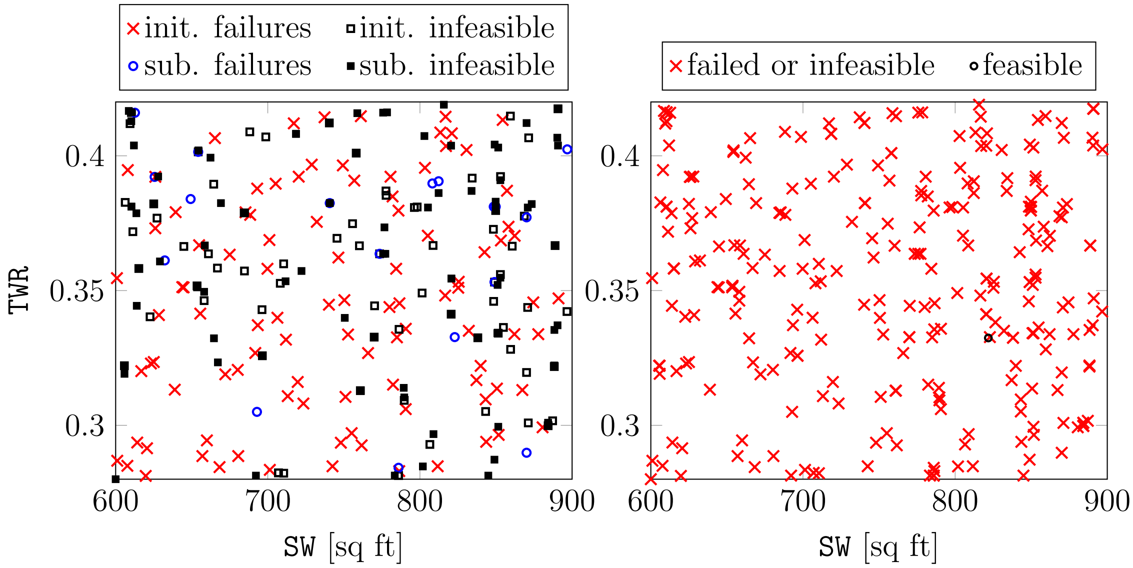


Figure 94: Sampling of GA.

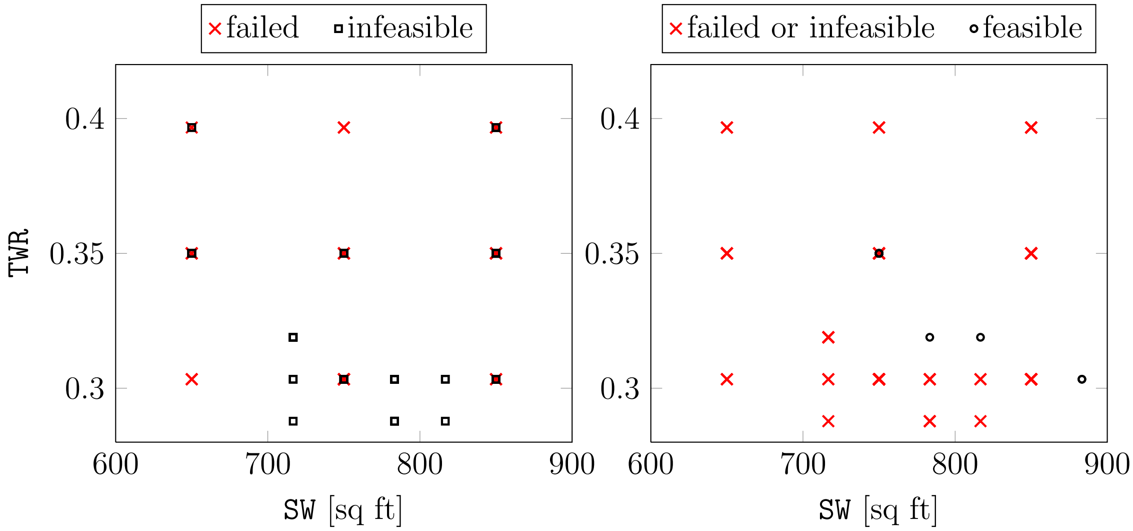


Figure 95: Sampling of DIRECT.

CHAPTER X

CONCLUSIONS

10.1 Summary Of Contributions

The primary contribution of this research is a fully Bayesian surrogate model-based global optimization algorithm for computationally expensive design problems that are subject to the set of technical challenges listed in §1.2. The algorithm is named `fBcEGO` for fully Bayesian constrained Efficient Global Optimization. `fBcEGO` shows significant improvement over current state-of-the-art algorithms for nonlinearly constrained problems by obtaining larger reductions in the function values for a given function evaluation budget.

A critical step of any surrogate model-based global optimization algorithm is the model fit step. Since the ISC is based directly on the model, a poor model fit results in poor decision making for the placement of subsequent iterates. It was shown in §2.2.4 that the current state-of-the-art model fitting technique (MLE) was inadequate in situations where function evaluations are scarce. A fully Bayesian GP model was derived which marginalizes the uncertainty of the hyperparameters into the model, thereby enabling better decision making.

A novel method for assigning the hyperparameter priors was presented in §2.2.6. This method exploits Bayesian penalization to automatically construct a prior, eliminating the dependence on a user-inputted or static distribution, which may be a poor assumption. A discrete prior is used, which reduces the analytical effort of `fBcEGO` by eliminating modes that provide a negligible contribution to the model. Furthermore, the use of a discrete prior results in an analytical expression for the model. This eliminates the need to marginalize the hyperparameters by some numerical integration method.

10.2 Revisiting The Research Questions & Hypotheses

The research questions and hypotheses are revisited in this section. In order to develop an algorithm that was more efficient than the current state-of-the-art, it was necessary to address not only the technical challenges, but also the technology gaps in the state-of-the-art methods. The primary research question asked what type of algorithm can outperform the state-of-the-art methods within the context of surrogate model-based global optimization. It was hypothesized that a fully Bayesian surrogate model-based global optimization algorithm that utilized a fully Bayesian EI criterion as the ISC would solve a larger percentage of black-box problems in fewer function evaluations than the state-of-the-art methods, and that the additional techniques employed to address the technical challenges would be adequate in retaining this performance for real-life design problems.

The primary research question was decomposed into a set of low-level research questions aimed at addressing the different components of `fBcEGO` in isolation. A key implied assumption throughout this research was that combining the best-performing components via a bottom-up approach would not result in any destructive interaction. The low-level research questions are restated here and addressed individually based on the results obtained from the algebraic test problems and aircraft-related applications.

Research Question 1: How does the initial design affect the performance of algorithms in terms of number of simplex gradients $N_{p,s}$ required to solve problems to within some accuracy ϵ?

It was found that for continuous bound constrained problems, the use of systematic designs resulted in inferior performance when compared with algorithms that employed an adaptive deterministic global solver (in this case, `DIRECT`) to generate the initial sample set. This adaptive initial sampling strategy enabled intelligent placement of samples by exploiting the information obtained from past samples. This is in contrast to static designs such as LHDs which evaluate the expensive function at fixed locations. Furthermore, it was found that use of smaller initial designs resulted in superior performance when compared with algorithms that employed larger designs. Smaller designs enabled the sophisticated surrogate model-based algorithms to place samples sooner, resulting in better performance for the same budget.

Research Question 2: Within the context of GP-based global optimization, what ISC has the highest potential to obtain the largest reduction in the function values of computationally expensive black-box problems under budget constraints? How does the performance of non-GP-based algorithms compare?

It was inferred from the observations in §2.2.4 that MLE-based algorithms were inferior to fully Bayesian algorithms. The fully Bayesian approach marginalizes the uncertainty of the hyperparameters into the surrogate model and hence ISC. This enables better decision making versus MLE when computing future iterates. By employing a weighted sum model with weights determined by the Bayesian methodology, more can be learned about the hyperparameters. It was found that for bound constrained problems, the fully Bayesian approach did not provide a significant performance gain over the state-of-the-art MLE-based algorithm EGO. A possible explanation for this result was that the bound constrained test problems were not difficult enough to scope the algorithms appropriately. While the performance of the fully Bayesian approach was better than or equal to the other GP-based approaches, its performance was dominated by CORS-RBF. This is a non-Bayesian approach which uses the surrogate model directly as the ISC, with some additional constraints to balance the global and local search. This evidence supports the notion that the bound constrained test problems were not difficult enough to appropriately determine the scope of the algorithms.

Research Question 3: Within the context of GP-based global optimization, how should nonlinear constraints be handled such that the resulting algorithm will solve more problems to a higher degree of accuracy given a budget? How does the performance of non-GP-based constraint-handling methods compare?

With the exception of a few outliers, `fBcEGO` demonstrated definitive superior performance on the algebraic nonlinearly constrained problems, solving up to three times as many problems than the competing algorithms for a given budget. `fBcEGO` uses the fully Bayesian version of the constrained EI criterion as the ISC. The multiplicative nature of the constrained EI criterion magnifies the poor decision-making abilities of the MLE approach when this method fails to generate a suitable model of the data. Nonsensical sample placement can result from the inadequacy of a single model. `DIRECT` is generally dominated by the surrogate model-based algorithms, all of which demonstrate similar performance with respect to each other.

Research Question 4: What strategy or strategies can be used to handle observations which have been corrupted by deterministic noise?

It was found that the regression-based EGO and `fBcEGO` algorithms were superior to either a regression-based P-algorithm or a regression-based one-stage EGO algorithm. No tests on noise corrupted problems were conducted for the non-GP-based algorithms.

Research Question 5: If a problem is believed to be nonsmooth, i.e., exhibits non-differentiable subspaces, how can this belief be included in a surrogate model-based algorithm? What if this belief is incorrect?

It was found that for the GP-based algorithms, employing nonsmooth basis functions to solve nonsmooth problems was a superior strategy versus employing smooth basis functions to solve the same problems. All nonsmooth problems were nondifferentiable at the global minimizers. It was inferred that if the global minimizer occurred in the differentiable region of the design space, employing nonsmooth basis functions would degrade performance. An unexpected result was that CORS-RBF performed worse on nonsmooth problems when employing a nonsmooth basis function. This is because the CORS framework directly uses the surrogate model as the ISC, thus the ability to accurately model the smooth regions becomes important. Performance also degrades if nonsmooth basis functions are employed to solve smooth problems. Nonsmooth basis functions should only be utilized in GP-based algorithms if there is a strong motivation to do so.

Research Question 6: When hard constraints are encountered, how should the failed values be imputed such that subsequent iterations are more likely to be successful?

`fBcEGO` was applied to an airfoil design problem with hard constraints. Three imputation methods were tested: predictor imputation, penalized imputation, and maximum value imputation. The penalized imputation method resulted in the best performance in terms of increased sampling in the feasible region and function value reduction. The penalized imputation method penalizes failed iterations by a penalty which increases as the distance to other samples increases. The magnitude of this penalty must be moderate such that the surrogate model through the feasible and imputed samples does not become warped. The predictor and maximum value imputation methods do not penalize failed iterations by a large enough penalty, which causes `fBcEGO` to sample more in the infeasible regions and in some cases to stall.

10.3 Recommendations

`fBcEGO` is not appropriate for all problems. Every algorithm has a specialized niche of problems that it is well-suited to solve. The No Free Lunch theorem for optimization states that "for any algorithm, any elevated performance over one class of problems is offset by performance over another class" [126]. The more technical challenges a problem exhibits, the more suitable `fBcEGO` is to solve it. However, the primary consideration is the computational expense. When problems become very expensive, the function evaluation budget is limited and there is no choice but to resort to sophisticated algorithms such as `fBcEGO`. Highly sophisticated algorithms have large analytical times; however, it is the *relative* analytical time when compared with a single function evaluation of the expensive code that is important. `fBcEGO` is especially well-suited for solving very expensive nonlinearly constrained problems, as indicated by the results in §8.4.

In theory, `fBcEGO` can be applied to problems of arbitrary dimension and with an arbitrary number of constraints, but in practice, the analytical time of the algorithm will need to be considered relative to the time required to make one call to the function being optimized. The proposed model fitting method bypasses the curse of dimensionality and is indirectly dependent on the dimension of the problem through the samples. The evaluation of the fully Bayesian EI criterion is also independent of the dimension of the problem, but requires $\mathcal{O}(mk^3)$ operations at each iteration

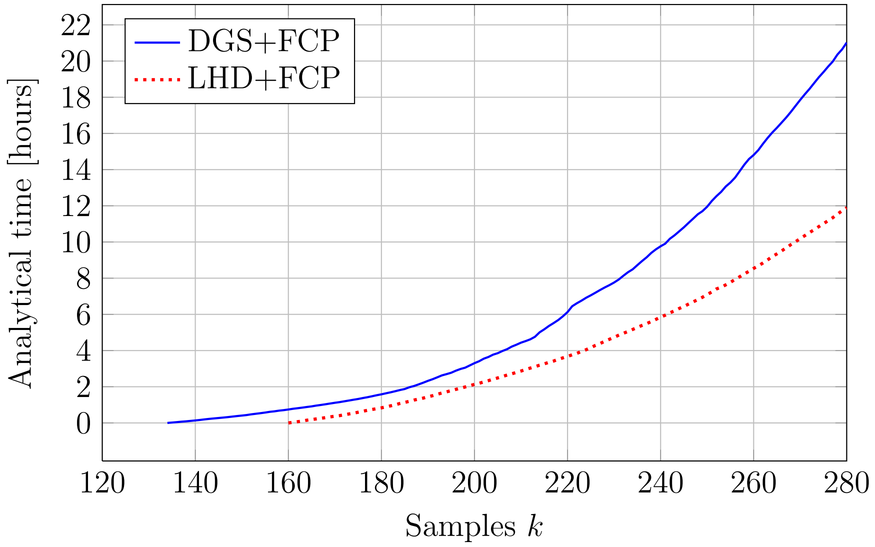


Figure 96: Number of samples k versus total analytical time of `fBcEGO` on FLOPS problem.

where m is the equivalent number of inequality constraints (recall that this is equal to $|\mathcal{I}| + 2|\mathcal{E}|$) and k is the number of samples. The maximization of the ISC depends on the dimension of the problem indirectly through the number of samples and directly through the method used to maximize the ISC. In this research, `DIRECT` was used with $100n$ function evaluations as the ISC subsolver. Other methods may scale differently with n.

As a benchmark, the aircraft design problem required approximately 12 hours to terminate with $N = 20$ simplex gradients ($12n + 3$ LHD+FCP samples included) on 64-bit Windows 7 PC with 4GB RAM and a quad-core Intel i5 750 processor running at 2.67 GHz; four instances of MATLAB 7.9 were running in parallel. This time measure is due entirely to the analytical effort of `fBcEGO`, since one FLOPS execution required less than one second to complete (see Figure 96). For inexpensive problems, the analytical time of `fBcEGO` dominates, but `fBcEGO` has been specifically developed for very expensive functions, where the analytical time is only a small percentage of the time required to make one call to the expensive function.

10.4 Suggestions For Further Research

`fBcEGO` can benefit from parallelization of the model fitting method. The objective function and constraints are constructed in exactly the same way. Thus, parallelizing the model fit step can reduce `fBcEGO`'s wall time. `fBcEGO` requires $\mathcal{O}(mk^3)$ operations at each iteration, where m is the equivalent number of inequality constraints and k is the number of samples. Recall that the use of isotropic basis functions eliminated the curse of dimensionality at the model fit step, but the number of function evaluations required by surrogate model-based global optimization algorithms to solve problems is still dependent on the dimension of the problem.

The primary solver (`DIRECT`) used to maximize the ISC for the competing algorithms can also benefit from parallelization. The constrained EI criterion requires $\mathcal{O}(Imk^3)$ operations at each iteration, where I is the number of elements in the priors $p(\theta)$ in the fully Bayesian approach. Maximization of the ISC can become expensive for larger problems. `DIRECT` makes one or more independent function evaluations of the constrained EI criterion at each iteration, with the first three evaluations always being the same. Thus, these evaluations can be parallelized.

It was mentioned in Remarks 3.2.1, 3.2.2, and 3.2.3 that the EI criterion can underflow to zero. This was a common situation during this research and not an isolated case. For the MLE-based criteria, it was possible to write an asymptotic expansion for the ISC to eliminate underflow. For the fully Bayesian approach this was not possible. The implementation of the student t distribution which defined the fully Bayesian EI criterion was more stable than the $\text{erf}(\cdot)$ function that defined the MLE-based EI criterion, but underflow and precision-related problems still occurred. A solution to this issue at the mathematical level may further improve the performance of `fBcEGO`.

APPENDIX A

TEST PROBLEMS

A.1 Bound Constrained Test Problems

B-1. Schubert problem [46]

$$\begin{aligned} \underset{x \in \mathbb{R}}{\text{minimize}} \quad & -\sum_{i=1}^{5} k \sin[(k+1)x + k] \\ \text{subject to} \quad & -10 \leq x \leq 10 \end{aligned} \tag{B-1}$$

There are three global minima located at $x^* = -6.774576$, $x^* = -6.774576 + 2\pi$, and $x^* = -6.774576 + 4\pi$ with $y(x^*) = -12.031249$. There are 16 additional nonglobal minima.

B-2. Branin function [29]

$$\begin{aligned} \underset{x \in \mathbb{R}^2}{\text{minimize}} \quad & \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos(x_1) + 10 \\ \text{subject to} \quad & -5 \leq x_1 \leq 10 \\ & 0 \leq x_2 \leq 15 \end{aligned} \tag{B-2}$$

There are three global minima at $x^* = (-\pi, 12.274999)$, $x^* = (\pi, 2.275000)$, and $x^* = (9.424777, 2.474999)$ with $y(x^*) = 5/(4\pi)$

B-3. Camel back 3 [5]

$$\begin{aligned} \underset{x \in \mathbb{R}^2}{\text{minimize}} \quad & 2x_1^2 - 1.05x_1^4 + \tfrac{1}{6}x_1^6 + x_1x_2 + x_2^2 \\ \text{subject to} \quad & -1.9 \leq x_1 \leq 1.9 \\ & -1.1 \leq x_2 \leq 1.1 \end{aligned} \tag{B-3}$$

There is one global minimum at $x^* = (0, 0)$ with $y(x^*) = 0$ and two other nonglobal minima.

B-4. Camel back 6 [5]

$$\begin{aligned}
\underset{x\in\mathbb{R}^2}{\text{minimize}} \quad & 4x_1^2 - 2.1x_1^4 + \tfrac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4 \\
\text{subject to} \quad & -1.9 \le x_1 \le 1.9 \\
& -1.1 \le x_2 \le 1.1
\end{aligned} \tag{B-4}$$

This function possesses 180° symmetry about the origin and has three conjugate pairs of minima. There is one pair of global minima at $x^* = (\pm 0.0898420, \mp 0.712656)$ with $y(x^*) = -1.031628$ and two additional pairs of nonglobal minima.

B-5. Dixon & Price function ($n = 2$)

$$\begin{aligned}
\underset{x\in\mathbb{R}^n}{\text{minimize}} \quad & (x_1 - 1)^2 + 2(2x_2^2 - x_2)^2 \\
\text{subject to} \quad & -1 \le x_1 \le 2 \\
& -1 \le x_2 \le 1
\end{aligned} \tag{B-5}$$

There is one global minimum at $x^* = (1, 1/\sqrt{2})$ with $y(x^*) = 0$ and one nonglobal minimum at $x^* = (1/3, 0)$ with $y(x^*) = 2/3$.

B-6. Goldstein & Price function [29]

$$\begin{aligned}
\underset{x\in\mathbb{R}^2}{\text{minimize}} \quad & [1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)] \\
& \times [30 + (2x_1 - 3x_2)^2(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)] \\
\text{subject to} \quad & -2 \le x_i \le 2, \quad i = 1, 2
\end{aligned} \tag{B-6}$$

This function has four nonglobal local minima and one global minimum at $x^* = (0, -1)$ with $y(x^*) = 3$.

B-7. n-dimensional Modified Langerman Problem [5]

$$\begin{aligned}
\underset{x\in\mathbb{R}^n}{\text{minimize}} \quad & -\sum_{j=1}^{5} c_j \cos(\pi d_j) \exp(-d_j/\pi) \\
\text{subject to} \quad & 0 \le x_i \le 10, \quad i = 1, \ldots, n
\end{aligned} \tag{B-7}$$

where $d_j = \sum_{i=1}^{n}(x_i - a_{ji})^2$. The coefficients c_j and a_{ji} are given in Table 31. The number of local minima is unknown. Some global minima are shown in Table 32.

Table 31: Data for n-dimensional Modified Langerman Problem, test problem B-7.

j	c_j	a_{ji}									
		$i = 1$	2	3	4	5	6	7	8	9	10
1	0.806	9.681	0.667	4.783	9.095	3.517	9.325	6.544	0.211	5.122	2.020
2	0.517	9.400	2.041	3.788	7.931	2.882	2.672	3.568	1.284	7.033	7.374
3	0.100	8.025	9.152	5.114	7.621	4.564	4.711	2.996	6.126	0.734	4.982
4	0.908	2.196	0.415	5.649	6.979	9.510	9.166	6.304	6.054	9.377	1.426
5	0.965	8.074	8.777	3.467	1.867	6.708	6.349	4.534	0.276	7.633	1.567

Table 32: Some global minimizers of the n-dimensional Modified Langerman Problem, test problem B-7.

n	2	5	7	10
$y(x^*)$	-1.030632	-0.965000	-0.517000	-0.965000

B-8. Modified Rosenbrock function [5]

$$\begin{aligned} &\underset{x\in\mathbb{R}^2}{\text{minimize}} \quad 100\left(x_1^2 - x_2\right)^2 + \left(6.4(x_2 - \tfrac{1}{2})^2 - x_1 - \tfrac{3}{5}\right)^2 \\ &\text{subject to} \quad -5 \le x_i \le 5, \quad i = 1, 2 \end{aligned} \tag{B-8}$$

This function has two global minima at $x^* = (1, 1)$ with $y(x^*) = 0$ and $x^* = (0.3412, 0.1164)$ with $y(x^*) = 0$

B-9. n-dimensional Paviani problem [5, 49]

$$\begin{aligned} &\underset{x\in\mathbb{R}^n}{\text{minimize}} \quad -\sum_{i=1}^{n}\left((\log x_i - 2)^2 + \log(10 - x_i)^2\right)^2 - \left(\prod_{i=1}^{n} x_i\right)^{0.2} \\ &\text{subject to} \quad 2.001 \le x_i \le 9.999, \quad i = 1, \ldots, n \end{aligned} \tag{B-9}$$

The number of local minima is unknown. The global minima are presented in Table 33.

Table 33: Global optimizers for n-dimensional Paviani problem, test problem B-9.

n	$y(x^*)$	x^*
2	4.981510	$(8.538791, 8.538791)$
5	9.730525	$(8.740704, \ldots, 8.740704)$
10	-45.778470	$(9.350257, \ldots, 9.350257)$

B-10. Gulf R&D problem [70]

$$\begin{aligned} \underset{x \in \mathbb{R}^3}{\text{minimize}} \quad & \sum_{i=1}^{99} \left[\exp\left(-\frac{(u_i - x_2)^{x_3}}{x_1} \right) - \frac{i}{100} \right]^2 \\ \text{subject to} \quad & 0.1 \leq x_1 \leq 100 \\ & 0 \leq x_2 \leq 25.6 \\ & 0 \leq x_3 \leq 5 \end{aligned} \tag{B-10}$$

where

$$u_i = 25 + \left(-50 \log \frac{i}{100} \right)^{2/3}$$

There is one global minimum at $x^* = (50, 25, 1.5)$ with $y(x^*) = 0$.

B-11. Hartman function ($n = 3, 6$) [29]

$$\begin{aligned} \underset{x \in \mathbb{R}^n}{\text{minimize}} \quad & -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{n} a_{ij} (x_j - p_{ij})^2 \right) \\ \text{subject to} \quad & 0 \leq x_i \leq 1, \quad i = 1, \ldots, n \end{aligned} \tag{B-11}$$

The parameters c_i, a_{ij}, and p_{ij}, $i = 1, \ldots, n, j = 1, 2, 3$, are given in Table 34. For $n = 3$, the global minimum is located at $x^* = (0.114614, 0.555649, 0.852547)$ with $y(x^*) = -3.862782$. For $n = 6$, the global minimum is located at

$$x^* = (0.201690, 0.150011, 0.476874, 0.275332, 0.311652, 0.657301)$$

with $y(x^*) = -3.322368$.

Table 34: Data for the Hartman function, test problem B-11.

(a) $n = 3$

i	a_{ij}			c_i	p_{ij}		
1	3.0	10	30	1.0	3689.0	117.0	2673.
2	0.1	10	35	1.2	0.4699	0.4387	0.747
3	3.0	10	30	3.0	1091.0	8732.0	5547.
4	0.1	10	35	3.2	0.03815	0.5743	0.8828

(b) $n = 6$

i	a_{ij}						c_i	p_{ij}					
1	10.00	3.0	17.00	3.5	1.7	8	1.0	0.1312	0.1696	0.5569	0.0124	0.8283	0.5886
2	0.05	10.0	17.00	0.1	8.0	14	1.2	0.2329	0.4135	0.8307	0.3736	0.1004	0.9991
3	3.00	3.5	1.70	10.0	17.0	8	3.0	0.2348	0.1451	0.3522	0.2883	0.3047	0.6650
4	17.00	8.0	0.05	10.0	0.1	14	3.2	0.4047	0.8828	0.8732	0.5743	0.1091	0.0381

B-12. n-dimensional Michalewicz function [12]

$$\begin{aligned} \underset{x\in\mathbb{R}^n}{\text{minimize}} \quad & -\sum_{i=1}^{n} \sin(x_i)\left(\sin(ix_i^2/\pi)\right)^{2m} \\ \text{subject to} \quad & 0 \le x_i \le \pi, \quad i = 1, 2, \ldots, n \end{aligned} \tag{B-12}$$

The function uses $m = 10$. The number of local minima is unknown. For $n = 5$, $y(x^*) = -4.687658$ and for $n = 10$, $y(x^*) = -9.660152$.

B-13. Shekel's Foxholes ($n = 2$) [5]

$$\begin{aligned} \underset{x\in\mathbb{R}^2}{\text{minimize}} \quad & -\sum_{j=1}^{30} \frac{1}{c_j + \sum_{i=1}^{n}(x_i - a_{ji})^2} \\ \text{subject to} \quad & 0 \le x_i \le 10, \quad i = 1, \ldots, n \end{aligned} \tag{B-13}$$

The parameters c_j and a_{ji}, $i = 1, 2, \ldots, n$, $j = 1, 2, \ldots, 30$ are given in Table 35. The global minimum is located at $x^* = (8.024065, 9.146534)$ with $y(x^*) = -12.119008$.

B-14. Shekel-m function

$$\begin{aligned} \underset{x\in\mathbb{R}^n}{\text{minimize}} \quad & -\sum_{j=1}^{m} \frac{1}{c_j + \sum_{i=1}^{n}(x_i - a_{ji})^2} \\ \text{subject to} \quad & 0 \le x_i \le 10, \quad i = 1, \ldots, n \end{aligned} \tag{B-14}$$

Table 35: Data for Shekel's Foxholes, test problem B-13.

j	c_j	a_{ji}									
		$i = 1$	2	3	4	5	6	7	8	9	10
1	0.806	9.681	0.667	4.783	9.095	3.517	9.325	6.544	0.211	5.122	2.020
2	0.517	9.400	2.041	3.788	7.931	2.882	2.672	3.568	1.284	7.033	7.374
3	0.100	8.025	9.152	5.114	7.621	4.564	4.711	2.996	6.126	0.734	4.982
4	0.908	2.196	0.415	5.649	6.979	9.510	9.166	6.304	6.054	9.377	1.426
5	0.965	8.074	8.777	3.467	1.863	6.708	6.349	4.534	0.276	7.633	1.567
6	0.669	7.650	5.658	0.720	2.764	3.278	5.283	7.474	6.274	1.409	8.208
7	0.524	1.256	3.605	8.623	6.905	0.584	8.133	6.071	6.888	4.187	5.448
8	0.902	8.314	2.261	4.224	1.781	4.124	0.932	8.129	8.658	1.208	5.762
9	0.531	0.226	8.858	1.420	0.945	1.622	4.698	6.228	9.096	0.972	7.637
10	0.876	7.305	2.228	1.242	5.928	9.133	1.826	4.060	5.204	8.713	8.247
11	0.462	0.652	7.027	0.508	4.876	8.807	4.632	5.808	6.937	3.291	7.016
12	0.491	2.699	3.516	5.874	4.119	4.461	7.496	8.817	0.690	6.593	9.789
13	0.463	8.327	3.897	2.017	9.570	9.825	1.150	1.395	3.885	6.354	0.109
14	0.714	2.132	7.006	7.136	2.641	1.882	5.943	7.273	7.691	2.880	0.564
15	0.352	4.707	5.579	4.080	0.581	9.698	8.542	8.077	8.515	9.231	4.670
16	0.869	8.304	7.559	8.567	0.322	7.128	8.392	1.472	8.524	2.277	7.826
17	0.813	8.632	4.409	4.832	5.768	7.050	6.715	1.711	4.323	4.405	4.591
18	0.811	4.887	9.112	0.170	8.967	9.693	9.867	7.508	7.770	8.382	6.740
19	0.828	2.440	6.686	4.299	1.007	7.008	1.427	9.398	8.480	9.950	1.675
20	0.964	6.306	8.583	6.084	1.138	4.350	3.134	7.853	6.061	7.457	2.258
21	0.789	0.652	2.343	1.370	0.821	1.310	1.063	0.689	8.819	8.833	9.070
22	0.360	5.558	1.272	5.756	9.857	2.279	2.764	1.284	1.677	1.244	1.234
23	0.369	3.352	7.549	9.817	9.437	8.687	4.167	2.570	6.540	0.228	0.027
24	0.992	8.798	0.880	2.370	0.168	1.701	3.680	1.231	2.390	2.499	0.064
25	0.332	1.460	8.057	1.336	7.217	7.914	3.615	9.981	9.198	5.292	1.224
26	0.817	0.432	8.645	8.774	0.249	8.081	7.461	4.416	0.652	4.002	4.644
27	0.632	0.679	2.800	5.523	3.049	2.968	7.225	6.730	4.199	9.614	9.229
28	0.883	4.263	1.074	7.286	5.599	8.291	5.200	9.214	8.272	4.398	4.506
29	0.608	9.496	4.830	3.150	8.270	5.079	1.231	5.731	9.494	1.883	9.732
30	0.326	4.138	2.562	2.532	9.661	5.611	5.500	6.886	2.341	9.699	6.500

The parameters c_j and a_{ji}, $i = 1, 2, \ldots, n$, $j = 1, 2, \ldots, m$ are given in Table 36. The number of local minima is unknown. The global minima for $n = 5,\ 7$, and 10 are presented in Table 37.

Table 36: Data for the Shekel-m function, test problem B-14.

j	c_j	a_{ji}			
		$i = 1$	2	3	4
1	0.1	4.0	4.0	4.0	4.0
2	0.2	1.0	1.0	1.0	1.0
3	0.2	8.0	8.0	8.0	8.0
4	0.4	6.0	6.0	6.0	6.0
5	0.4	3.0	7.0	3.0	7.0
6	0.6	2.0	9.0	2.0	9.0
7	0.3	5.0	5.0	3.0	3.0
8	0.7	8.0	1.0	8.0	1.0
9	0.5	6.0	2.0	6.0	2.0
10	0.5	7.0	3.6	7.0	3.6

Table 37: Some global minimizers for the Shekel-m function, test problem B-14.

m	$y(x^*)$	x^*
5	-10.153200	$(4, 4, 4, 4)$
7	-10.402941	$(4, 4, 4, 4)$
10	-10.536410	$(4, 4, 4, 4)$

A.2 Nonlinearly Constrained Test Problems

N-1. Constrained Schubert problem 1

$$\begin{aligned} \underset{x\in\mathbb{R}}{\text{minimize}} \quad & -\sum_{i=1}^{5} k\sin[(k+1)x+k] \\ \text{subject to} \quad & 10\cos(x)\exp\left(-0.5(x/\sqrt{10})^2\right) - x \geq 0 \\ & -10 \leq x \leq 10 \end{aligned} \tag{B-1}$$

There are two global minima at $x^* = -6.774576$ and $x^* = -6.774576 + 2\pi$ with $y(x^*) = -12.031249$. There are nine additional nonglobal minima.

N-2. Constrained Schubert problem 2

$$\begin{aligned} \underset{x\in\mathbb{R}}{\text{minimize}} \quad & -\sum_{i=1}^{5} k\sin[(k+1)x+k] \\ \text{subject to} \quad & 10\cos(x)\exp\left(-0.5(x/\sqrt{10})^2\right) - x \geq 0 \\ & \sin(\pi x) \geq 0 \\ & -10 \leq x \leq 10 \end{aligned} \tag{B-2}$$

There is one global minimum at $x^* = -1.725492$ with $y(x^*) = -9.494706$ and twelve additional nonglobal minima.

N-3. Hock & Schittkowski 9 [49]

$$\begin{aligned} \underset{x\in\mathbb{R}^2}{\text{minimize}} \quad & \sin(\pi x_1/12)\cos(\pi x_2/16) \\ \text{subject to} \quad & 4x_1 - 3x_2 = 0 \\ & 4x_1 - 3x_2 \geq 0 \\ & -15 \leq x_i \leq 15, \quad i = 1, 2 \end{aligned} \tag{B-3}$$

There are two global minima at $x^* = (-3, -4)$ and $x^* = (9, 12)$ with $y(x^*) = -1/2$.

N-4. Multiple disconnected regions

$$\begin{aligned} \underset{x\in\mathbb{R}^2}{\text{minimize}} \quad & x_1 - x_2 \\ \text{subject to} \quad & \sin(\pi x_1)\cos(\pi x_2) - 0.5 \geq 0 \\ & -2 \leq x_i \leq 2, \quad i = 1,2 \end{aligned} \tag{B-4}$$

This problem features ten feasible disconnected convex regions with a linear objective function. Hence, there are a total of ten minima, one of which is global at $x^* = (\pi^{-1}\sin(0.5) - 2, 2)$ with $y(x^*) = \pi^{-1}\sin(0.5) - 4$.

N-5. Hock & Schittkowski 12 [49]

$$\begin{aligned} \underset{x\in\mathbb{R}^2}{\text{minimize}} \quad & 0.5x_1^2 + x_2^2 - x_1x_2 - 7x_1 - 7x_2 \\ \text{subject to} \quad & 25 - 4x_1^2 - x_2^2 \geq 0 \\ & -5 \leq x_i \leq 5, \quad i = 1,2 \end{aligned} \tag{B-5}$$

There is global one minimum at $x^* = (2,3)$ with $y(x^*) = -30$.

N-6. Hock & Schittkowski 14 [49]

$$\begin{aligned} \underset{x\in\mathbb{R}^2}{\text{minimize}} \quad & (x_1 - 2)^2 + (x_2 - 1)^2 \\ \text{subject to} \quad & -0.25x_1^2 - x_2^2 + 1 \geq 0 \\ & x_1 - 2x_2 + 1 = 0 \\ & -5 \leq x_i \leq 5, \quad i = 1,2 \end{aligned} \tag{B-6}$$

There is one global minimum at $x^* = (1/2(\sqrt{7} - 1), 1/4(\sqrt{7} + 1))$ with $y(x^*) = 9 - 2.875\sqrt{7}$.

N-7. Hock & Schittkowski 19 [49]

$$\begin{aligned} \underset{x\in\mathbb{R}^2}{\text{minimize}} \quad & (x_1 - 10)^3 + (x_2 - 20)^3 \\ \text{subject to} \quad & (x_1 - 5)^2 + (x_2 - 5)^2 - 100 \geq 0 \\ & -(x_2 - 5)^2 - (x_1 - 6)^2 + 82.81 \geq 0 \\ & 0 \leq x_i \leq 20, \quad i = 1,2 \end{aligned} \tag{B-7}$$

There is one global minimum at $x^* = (14.095, 0.842961)$ with $y(x^*) = -6961.813875$.

N-8. Mystery function [106]

$$\begin{aligned} \underset{x\in\mathbb{R}^2}{\text{minimize}} \quad & 2 + 0.01(x_2 - x_1^2)^2 + (1 - x_1)^2 \\ & + 2(2 - x_2)^2 + 7\sin(0.5x_1)\sin(0.7x_1x_2) \\ \text{subject to} \quad & \sin(x_1 - x_2 - \pi/8) \geq 0 \\ & 0 \leq x_i \leq 5, \quad i = 1, 2 \end{aligned} \tag{N-8}$$

There is one global minimum at $x^* = (2.744951, 2.352252)$ with $y(x^*) = -1.174274$.

The number of local minima is unknown.

N-9. Hock & Schittkowski 26 [49]

$$\begin{aligned} \underset{x\in\mathbb{R}^3}{\text{minimize}} \quad & (x_1 - x_2)^2 + (x_2 - x_3)^4 \\ \text{subject to} \quad & (1 + x_2^2)x_1 + x_3^4 - 3 = 0 \\ & -5 \leq x_i \leq 5, \quad i = 1, \ldots, 3 \end{aligned} \tag{B-9}$$

There are two global minima at $x^* = (1, 1, 1)$ and $x^* = (a, a, a)$ with $y(x^*) = 0$.

The constant a is given as

$$a = \sqrt[3]{\alpha - \beta} - \sqrt[3]{\alpha + \beta} - 2/3 \tag{A.1}$$

$$\alpha = \sqrt{139/108} \tag{A.2}$$

$$\beta = 61/54 \tag{A.3}$$

N-10. Hock & Schittkowski 32 [49]

$$\begin{aligned} \underset{x\in\mathbb{R}^3}{\text{minimize}} \quad & (x_1 + 3x_2 + x_3)^2 + 4(x_1 - x_2)^2 \\ \text{subject to} \quad & 6x_2 + 4x_3 - x_1^3 \geq 0 \\ & 1 - x_1 - x_2 - x_3 = 0 \\ & 0 \leq x_i \leq 2, \quad i = 1, \ldots, 3 \end{aligned} \tag{B-10}$$

There is one global minimum at $x^* = (0, 0, 1)$ with $y(x^*) = 1$. The number of nonglobal minima is unknown.

N-11. Hock & Schittkowski 33 [49]

$$\begin{aligned} \underset{x\in\mathbb{R}^3}{\text{minimize}} \quad & (x_1-1)(x_1-2)(x_1-3)+x_3 \\ \text{subject to} \quad & x_3^2 - x_2^2 - x_1^2 \geq 0 \\ & x_1^2 + x_2^2 + x_3^2 - 4 \geq 0 \\ & 0 \leq x_i \leq 5, \qquad i = 1, \ldots, 3 \end{aligned} \tag{B-11}$$

There is one global minimum at $x^* = (0, \sqrt{2}, \sqrt{2})$ with $y(x^*) = \sqrt{2} - 6$. The number of nonglobal minima is unknown.

N-12. Phase and chemical equilibrium problem, binary system of hydrogen sulfide and methane [37, §8.6.7]

$$\begin{aligned} \underset{x\in\mathbb{R}^2, z\in\mathbb{R}}{\text{minimize}} \quad & \sum_{i=1}^{2} x_i \log x_i + \sum_{i=1}^{2} x_i \log \hat{\phi}_i - \sum_{i=1}^{2} x_i \log x_i^F \hat{\phi}_i^F \\ \text{subject to} \quad & \sum_{i=1}^{2} x_i \log \hat{\phi}_i = z - 1 - \log(z-B) - \frac{A}{B} \log\left(1 + \frac{B}{z}\right) \\ & z^3 - z^2 + (A - B^2 - B)z - AB = 0 \\ & \sum_{i=1}^{2} x_i = 1 \\ & 0 \leq x_i \leq 1, \qquad i = 1, 2 \\ & 0 \leq z \leq 1 \end{aligned} \tag{N-12}$$

where

$$A = \sum_{i=1}^{2} \sum_{j=1}^{2} \alpha_{ij} x_i x_j \tag{A.4}$$

$$B = \sum_{i=1}^{2} b_i x_i \tag{A.5}$$

The data for this problem is given below:

$$x^F = [0.0187, 0.9813]^T \tag{A.6}$$

$$\log \phi_i^F = [-1.0672, -0.3480]^T \tag{A.7}$$

$$b = [0.0771517, 0.0765784]^T \tag{A.8}$$

$$\alpha = \begin{bmatrix} 10.4633 & 0.579822 \\ 0.579822 & 0.379615 \end{bmatrix} \tag{A.9}$$

Because of the term $\log(z - B)$, when $z < B$, the objective function is set to a large positive (here, 10000) to drive the search away from this region. This problem has four nonglobal local minima and one global minimum at $(x^*, z^*) = (1, 0, 0.078314)$ with $y(x^*, z^*) = -82.112243$.

N-13. Test Problem 4 [36, §3.4]

$$\begin{aligned} \underset{x \in \mathbb{R}^3}{\text{minimize}} \quad & -2x_1 + x_2 - x_3 \\ \text{subject to} \quad & 4 - x_1 - x_2 - x_3 \geq 0 \\ & 6 - 3x_2 - x_3 \geq 0 \\ & x^T B^T B x - 2r^T B x + \|r\|_2^2 - 0.25\|b - v\|_2^2 \geq 0 \\ & 0 \leq x_1 \leq 2 \\ & 0 \leq x_2 \leq 2 \\ & 0 \leq x_3 \leq 3 \end{aligned} \tag{B-13}$$

where B is the following 3×3 matrix:

$$\begin{pmatrix} 0 & 0 & 1 \\ 0 & -1 & 0 \\ -2 & 1 & -1 \end{pmatrix}$$

$$b = [3, 0, -4]^T$$

$$v = [0, -1, -6]^T$$

$$r = [1.5, -0.5, -5]^T$$

There is one global minimum at $x^* = (0.5, 0, 3)$ with $y(x^*) = -4$.

N-14. Hock & Schittkowski 46 [49]

$$\begin{aligned} \underset{x\in\mathbb{R}^5}{\text{minimize}} \quad & (x_1 - x_2)^2 + (x_3 - 1)^2 + (x_4 - 1)^4 + (x_5 - 1)^6 \\ \text{subject to} \quad & x_1^2 x_4 + \sin(x_4 - x_5) - 1 = 0 \\ & x_2 + x_3^4 x_4^2 - 2 = 0 \\ & 0 \le x_i \le 3, \quad i = 1, \dots, 5 \end{aligned} \tag{B-14}$$

There is one global minimum at $x^* = (1, 1, 1, 1, 1)$ with $y(x^*) = 0$.

N-15. Hock & Schittkowski 81 [49]

$$\begin{aligned} \underset{x\in\mathbb{R}^5}{\text{minimize}} \quad & \exp(x_1 x_2 x_3 x_4 x_5) - 0.5(x_1^3 + x_2^3 + 1)^2 \\ \text{subject to} \quad & x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 - 10 = 0 \\ & x_2 x_3 - 5 x_4 x_5 = 0 \\ & x_1^3 + x_2^3 + 1 = 0 \\ & -2.3 \le x_i \le 2.3, \quad i = 1, 2 \\ & -3.2 \le x_i \le 3.2, \quad i = 3, 4, 5 \end{aligned} \tag{B-15}$$

There is one known solution at

$$x^* = (-1.717142, 1.159571, 1.827248, -0.7636474, -0.7636390)$$

with $y(x^*) = 0.0539498478$.

N-16. Hock & Schittkowski 83 [49]

$$\begin{aligned}
\underset{x\in\mathbb{R}^5}{\text{minimize}} \quad & 5.3578547x_3^2 + 0.8356891x_1x_5 \\
& +37.293239x_1 - 40792.141 \\
\text{subject to} \quad & a_1 + a_2x_2x_5 + a_3x_1x_4 - a_4x_3x_5 \geq 0 \\
& -a_1 - a_2x_2x_5 - a_3x_1x_4 + a_4x_3x_5 + 92 \geq 0 \\
& a_5 + a_6x_2x_5 + a_7x_1x_2 + a_8x_3^2 - 90 \geq 0 \\
& -a_5 - a_6x_2x_5 - a_7x_1x_2 - a_8x_3^2 + 90 + 20 \geq 0 \\
& a_9 + a_{10}x_3x_5 + a_{11}x_1x_3 + a_{12}x_3x_4 - 20 \geq 0 \\
& -a_9 - a_{10}x_3x_5 - a_{11}x_1x_3 - a_{12}x_3x_4 + 25 \geq 0 \\
& 78 \leq x_1 \leq 102 \\
& 33 \leq x_2 \leq 45 \\
& 27 \leq x_3 \leq 45 \\
& 27 \leq x_4 \leq 45 \\
& 27 \leq x_5 \leq 45
\end{aligned} \tag{N-16}$$

The coefficients a_i are given in Table 38. The number of local minima is unknown. The best known minimum occurs at $x^* = (78, 33, 29.9953, 45, 36.7758)$ with $y(x^*) = -30665.538672$.

Table 38: Data for test problem N-16.

i	a_i	i	a_i
1	85.334407	7	0.0029955
2	0.0056858	8	0.0021813
3	0.0006262	9	9.300961
4	0.0022053	10	0.0047026
5	80.51249	11	0.0012547
6	0.0071317	12	0.0019085

N-17. Continuous stirred tank reactor (CSTR) sequence design [37, §7.2.2]

$$\begin{aligned}
\underset{x\in\mathbb{R}^6}{\text{minimize}} \quad & -x_4 \\
\text{subject to} \quad & x_1 + k_1 x_1 x_5 = 1 \\
& x_2 - x_1 + k_2 x_2 x_6 = 0 \\
& x_3 + x_1 + k_3 x_3 x_5 = 1 \\
& x_4 - x_3 + x_2 - x_1 + k_4 x_4 x_6 = 0 \\
& \sqrt{x_5} + \sqrt{x_6} \le 4 \\
& \begin{array}{ccccc}
0 & \le & x_1 & \le & 1 \\
0 & \le & x_2 & \le & 1 \\
0 & \le & x_3 & \le & 1 \\
0 & \le & x_4 & \le & 1 \\
10^{-5} & \le & x_5 & \le & 16 \\
10^{-5} & \le & x_6 & \le & 16
\end{array}
\end{aligned} \tag{N-17}$$

where $k_1 = 0.09755988$, $k_2 = 0.99k_1$, $k_3 = 0.0391908$, $k_4 = 0.9$. The global minimum occurs at

$$x^* = (0.390855, 0.390855, 0.374613, 0.374610, 15.974712, 10^{-5})$$

with $y(x^*) = -0.374610$.

N-18. Hesse function [37, §3.4]

$$\begin{aligned}
\underset{x\in\mathbb{R}^6}{\text{minimize}} \quad & -25(x_1-2)^2-(x_2-2)^2-(x_3-1)^2 \\
& -(x_4-4)^2-(x_5-1)^2-(x_6-4)^2 \\
\text{subject to} \quad & (x_3-3)^2+x_4-4 \ \geq\ 0 \\
& (x_5-3)^2+x_6-4 \ \geq\ 0 \\
& 2-x_1+3x_2 \ \geq\ 0 \\
& 2+x_1-x_2 \ \geq\ 0 \\
& 6-x_1-x_2 \ \geq\ 0 \\
& x_1+x_2-2 \ \geq\ 0 \\
& 0 \leq x_1 \leq 6 \\
& 0 \leq x_2 \leq 2 \\
& 1 \leq x_3 \leq 5 \\
& 0 \leq x_4 \leq 6 \\
& 1 \leq x_5 \leq 5 \\
& 0 \leq x_6 \leq 10
\end{aligned} \tag{N-18}$$

This function is a concave quadratic with 18 local minima, one of which is global. The problem is separable into three two-dimensional problems in (x_1, x_2), (x_3, x_4), and (x_5, x_6). The global minimum is located at $x^* = (5, 1, 5, 0, 5, 10)$ with $y(x^*) = -310$.

N-19. Hock & Schittkowski 87 [49]

$$\begin{aligned}
\underset{x\in\mathbb{R}^6}{\text{minimize}} \quad & y_1(x)+y_2(x) \\
\text{subject to} \quad & 300-x_1-a_1^{-1}x_3x_4\cos(a_2-x_6)+a_3a_1^{-1}a_4x_3^2=0 \\
& -x_2-a_1^{-1}x_3x_4\cos(a_2+x_6)+a_3a_1^{-1}a_4x_4^2=0 \\
& -x_5-a_1^{-1}x_3x_4\sin(a_2+x_6)+a_3a_1^{-1}a_5x_4^2=0 \\
& 200-a_1^{-1}x_3x_4\sin(a_2-x_6)+a_3a_1^{-1}a_5x_3^2=0
\end{aligned} \tag{N-19}$$

$$\begin{array}{rcccl} 0 & \leq & x_1 & \leq & 400 \\ 0 & \leq & x_2 & \leq & 1000 \\ 340 & \leq & x_3 & \leq & 420 \\ 340 & \leq & x_4 & \leq & 420 \\ -1000 & \leq & x_5 & \leq & 10000 \\ 0 & \leq & x_6 & \leq & 0.5236 \end{array}$$

The coefficients a_i are given in Table 39 and the functions $y_1(\cdot)$ and $y_2(\cdot)$ are defined in equation (A.10). The number of local minima is unknown. The best known minimum occurs at $x^* = (204.201838, 99.999919, 383.252591, 419.999803, -11.490087, 0.072154)$ with $y(x^*) = 8926.052867$.

Table 39: Data for test problem N-19.

i	a_i
1	131.078
2	1.48577
3	0.90798
4	cos 1.47588
5	sin 1.47588

$$y_1(x) = \begin{cases} 30x_1, & 0 \leq x_1 < 300 \\ 31x_1, & 300 \leq x_1 \leq 400 \end{cases} \tag{A.10a}$$

$$y_2(x) = \begin{cases} 28x_2, & 0 \leq x_2 < 100 \\ 29x_2, & 100 \leq x_1 < 200 \\ 30x_2, & 200 \leq x_2 \leq 1000 \end{cases} \tag{A.10b}$$

N-20. Hock & Schittkowski 100 [49]

$$\begin{aligned}
\underset{x\in\mathbb{R}^7}{\text{minimize}} \quad & (x_1-10)^2+5(x_2-12)^2+x_3^4+3(x_4-11)^2+10x_5^6 \\
& +7x_6^2+x_7^4-4x_6x_7-10x_7-10x_6-8x_7 \\
\text{subject to} \quad & 127-2x_1^2-3x_2^4-x_3-4x_4^2-5x_5\geq 0 \\
& 282-7x_1-3x_2-10x_3^2-x_4+x_5\geq 0 \\
& 196-23x_1-x_2^2-6x_6^2+8x_7\geq 0 \\
& -4x_1^2-x_2^2+3x_1x_2-2x_3^2-5x_6+11x_7\geq 0 \\
& \begin{array}{rcccl}
1 & \leq & x_1 & \leq & 3 \\
1 & \leq & x_2 & \leq & 3 \\
-1 & \leq & x_3 & \leq & 1 \\
3 & \leq & x_4 & \leq & 5 \\
-1 & \leq & x_5 & \leq & 1 \\
0 & \leq & x_6 & \leq & 2 \\
0 & \leq & x_7 & \leq & 2
\end{array}
\end{aligned} \qquad \text{(N-20)}$$

The original version of this problem has no bound constraints but these are added for modeling purposes. There is one global minimum at

$$x^*=(2.330499,1.951372,-0.477541,4.365726,-0.624487,1.038131,1.594227)$$

with $y(x^*)=680.630057$.

A.3 Nonsmooth Problems

This section lists the nonsmooth test problems. Unless otherwise noted, the objective function for each problem is

$$y(x) = \sum_{i=1}^{m_p} |f_i(x)| \tag{A.11}$$

where m_p is the number of terms $f_i(x)$, which are given in each problem along with the bound constraints.

NS-1. One-dimensional nonsmooth problem

$$\begin{aligned} \underset{x \in \mathbb{R}}{\text{minimize}} \quad & y(x) \\ \text{subject to} \quad & 0 \leq x \leq 50 \end{aligned} \tag{NS-1}$$

where

$$y(x) = \begin{cases} -x, & x \leq 0 \\ -\frac{1}{5}x, & 0 < x \leq 5 \\ -\frac{1}{2}x + 1.5, & 5 < x \leq 10 \\ \frac{1}{4}x^2 - 28.5, & 10 < x \leq 20 \\ -2x + 112.5, & 20 < x \leq 25 \\ 8x - 137.5, & 25 < x \leq 33 \\ -x + 159.5, & 33 < x \leq 40 \\ x + 79.5, & 40 < x \leq 45 \\ -5x + 350.5, & 45 < x \leq 50 \\ x + 50.5, & 50 < x \end{cases} \tag{A.12}$$

There is one global minimum at $x^* = 10$ with $y(x^*) = -3.5$. There are three additional nonglobal minima.

NS-2. Two-dimensional nonsmooth problem 1

$$
\begin{aligned}
f_1(x) &= \xi_1 \\
f_2(x) &= \xi_2 \\
f_3(x) &= 2\xi_1^2 - 1.05\xi_1^4 + \tfrac{1}{6}\xi_1^6 + \xi_1\xi_2 + \xi_2^2 \\
&- 1.9 \leq \xi_1 \leq 1.9 \\
&- 1.1 \leq \xi_2 \leq 1.1
\end{aligned}
\tag{NS-2}
$$

Here, $\xi_1 = x_1 - 0.5$ and $\xi_2 = x_2 + 0.25$. There is one global minimum at $x^* = (0.5, -0.25)$ with $y(x^*) = 0$.

NS-3. Two-dimensional nonsmooth problem 2

$$
\begin{aligned}
f_1(x) &= \xi_1 \\
f_2(x) &= \xi_2 \\
f_3(x) &= 4\xi_1^2 - 2.1\xi_1^4 + \tfrac{1}{3}\xi_1^6 + \xi_1\xi_2 - 4\xi_2^2 + 4\xi_2^4 \\
&- 1.9 \leq \xi_1 \leq 1.9 \\
&- 1.1 \leq \xi_2 \leq 1.1
\end{aligned}
\tag{NS-3}
$$

Here, $\xi_1 = x_1 - 0.5$ and $\xi_2 = x_2 + 0.25$. There is one global minimum at $x^* = (0.5, -0.25)$ with $y(x^*) = 0$. There are four additional nonglobal minima.

NS-4. Two-dimensional nonsmooth problem 3

$$
\begin{aligned}
f_1(x) &= x_1 \\
f_2(x) &= 3 - \sum_{i=1}^{4} c_i \exp\left(- \sum_{j=1}^{2} a_{ij}(x_j - p_{ij})^2 \right) \\
&0 \leq x_i \leq 1, \qquad i = 1, \ldots, 3
\end{aligned}
\tag{NS-4}
$$

The term $f_2(x)$ is part of the Hartman family of functions (see test problem B-11). The parameters c_i, a_{ij}, and p_{ij}, $i, j = 1, 2, 3$, are given in Table 34. There is one global minimum at $x^* = (0.004109, 0.294240)$ with $y(x^*) = 0.004109$ and one nonglobal minimum at $x^* = (0, 1)$ with $y(x^*) = 0.0378342$.

NS-5. Two-dimensional nonsmooth problem 4

$$\begin{aligned} f_1(x) &= x_1 \\ f_2(x) &= 2x_2 \\ f_3(x) &= (x_1 - 1)^2 + 2(2x_2^2 - x_1)^2 - 5 \\ &- 1 \le x_1 \le 2 \\ &- 1 \le x_2 \le 1 \end{aligned} \tag{NS-5}$$

There is one global minimum at $x^* = (-0.868517, 0)$ with $y(x^*) = 0.868580$. There is one additional nonglobal minima at $x^* = (1.535183, 0)$ with $y(x^*) = 1.535190$.

NS-6. Two-dimensional nonsmooth problem 5

$$\begin{aligned} \underset{x \in \mathbb{R}^2}{\text{minimize}} \qquad & 10 + \left(x_2 - \frac{5.1}{4\pi^2} x_1^2 + \frac{5}{\pi} x_1 - 6 \right)^2 \\ & + 10 \left(1 - \frac{1}{8\pi} \right) \cos(x_1) + 10\,|x_1| + 10\,|x_2| \\ \text{subject to} \quad & -10 \le x_i \le 15, \qquad i = 1, 2 \end{aligned} \tag{NS-6}$$

There is one global minimum at $x^* = (2.597988, 0)$ with $y(x^*) = 3.562570$.

NS-7. Two-dimensional nonsmooth problem 6

$$\begin{aligned} f_1(x) &= 0.1x_1 - 4 \\ f_2(x) &= 0.1x_2 - 5 \\ f_3(x) &= 1 - \sum_{j=1}^{30} \frac{1}{c_j + \sum_{i=1}^{n} (x_i - a_{ji})^2} \\ &5 \le x_i \le 12, \qquad i = 1, 2 \end{aligned} \tag{NS-7}$$

The parameters c_j and a_{ji}, $i = 1, 2$, $j = 1, \ldots, 30$ are given in Table 35. There is one global minimum at $x^* = (8.669134, 9.626276)$ with $y(x^*) = 7.170459$. The number of nonglobal minima is unknown.

NS-8. Two-dimensional nonsmooth problem 7

$$\begin{aligned} f_1(x) &= x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6 \\ f_2(x) &= 10\left(1 - \frac{1}{8\pi}\right)\cos(x_1/\pi) \\ f_3(x) &= 10 \\ & -15 \leq x_i \leq 15, \qquad i = 1, 2 \end{aligned} \tag{NS-8}$$

There are two global minima at $x^* = (4.934802, 1.291955)$ and $x^* = (14.804407, 10.751483)$ with $y(x^*) = 10$. There is one local minimum at $x^* = (-4.934802, 14.900335)$ with $y(x^*) = 12.099583$.

NS-9. Nonsmooth Hartman 3 problem

$$\begin{aligned} f_1(x) &= -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{3} a_{ij}(x_j - p_{ij})^2\right) \\ & 0 \leq x_i \leq 1, \qquad i = 1, 2, 3 \end{aligned} \tag{NS-9}$$

This problem has been derived by taking the absolute value of the Hartman function, test problem B-11, with $n = 3$. The parameters c_i, a_{ij}, and p_{ij}, $i = 1, \ldots, n, j = 1, 2, 3$, are given in Table 34. There is one global minimum at $x^* = (1, 1, 0)$ with $y(x^*) = 3.772719 \cdot 10^{-5}$. The number of local minima is unknown.

APPENDIX B

BAYESIAN MONTE CARLO

Bayesian Monte Carlo (BMC) [76, 93] is a method for performing Bayesian inference about the value of a nonanalytic integral

$$\overline{y}_p = \int y(x)p(x)dx \tag{B.1}$$

where $p(x)$ is the probability density of x and $y(x)$ is the function to be integrated. Simple Monte Carlo (SMC) makes the approximation

$$\overline{y}_p \simeq \frac{1}{k}\sum_{i=1}^{k} y(x^{(i)}) \tag{B.2}$$

where $x^{(i)}$ are random draws from $p(x)$, which converges to the right answer in the limit of large numbers of samples k. Thus, the first objection to SMC is the computational inefficiency of the method. The second objection is that SMC procedures ignore the values of $x^{(i)}$ when forming the estimate, e.g., if three points are sampled and two happen to be the same point (thereby conveying no new information about the integrand), then averaging the samples is clearly inappropriate.

Considering the evaluation of the integral (B.1) as a Bayesian inference problem avoids the inconsistencies of the SMC approach and can result in better estimates with fewer function evaluations. The unknown quantity $\overline{y}_p$ can be considered as a random variable and the uncertainty in $\overline{y}_p$ arises because $y(x)$ cannot be evaluated for every x. The strategy is to use Bayes' theorem: assign a prior to $y(x)$ and condition the prior on the observations $y^{(k)} = [y(x^{(1)}), \ldots, y(x^{(k)})]^T$. Since the integral (B.1) is a linear projection on the direction defined by $p(x)$, the posterior over $\overline{y}_p$ is obtained by integrating equation (B.1) with the proper substitutions.

A convenient way of putting priors over functions is through GPs. The classical assumption is a zero mean, noise-free GP prior, but BMC is extended here to account for a constant mean, noise corrupted prior. Under a constant mean GP prior, the joint distribution of any finite number of noise corrupted samples is Gaussian:

$$y^{(k)} \sim \mathcal{N}(\beta, \mathbf{\Psi} \triangleq \mathbf{K} + \theta_n \mathbf{I}) \tag{B.3}$$

The elements of the covariance matrix $\mathbf{K}$ are given by a covariance function, a convenient choice being

$$\begin{aligned} \mathbb{k}(x^{(i)}, x^{(j)}) &= \theta_0 \exp\left[-\frac{1}{2} \sum_{h=1}^{n} \left(\frac{x_h^{(i)} - x_h^{(j)}}{\theta_h} \right)^2 \right] \\ &= \theta_0 \exp\left[-\frac{1}{2} \left(x^{(i)} - x^{(j)} \right)^T A^{-1} \left(x^{(i)} - x^{(j)} \right) \right] \end{aligned} \tag{B.4}$$

where $A = \operatorname{diag}(\theta_1^2, \dots, \theta_n^2)$ and θ_0, θ_h are the hyperparameters. BMC assigns a prior $p(y)$ to y and updates the posterior $p(y|y^{(k)})$ after making the observations $y^{(k)} = [y(x^{(1)}), \dots, y(x^{(k)})]$. The posterior $p(\overline{y}_p|y^{(k)})$ is also Gaussian and its mean and variance are given by [93]:

$$\begin{aligned} \mathbb{E}_{\overline{y}_p}(x) &= \int \left[\int y(x) p(y|y^{(k)}) dy \right] p(x) dx \\ &= \int \mathbb{E}[y|y^{(k)}] p(x) dx = \int \hat{y}(x) p(x) dx \end{aligned} \tag{B.5a}$$

$$\begin{aligned} \operatorname{var}(\overline{y}_p) &= \iint \left[\int \left[y(x) - \mathbb{E}_y(x) \right] \left[y(x_0) - \mathbb{E}_y(x_0) \right] p(y|y^{(k)}) dy \right] p(x) p(x_0) dx dx_0 \\ &= \iint \operatorname{Cov}(x, x_0) p(x) p(x_0) dx dx_0 = \iint s^2(x) p(x) p(x_0) dx dx_0 \end{aligned} \tag{B.5b}$$

where $\mathbb{E}[y]$ and $\operatorname{Cov}(x, x_0)$ are the posterior mean and covariance functions of y, respectively. The standard results for the predictive posterior mean and covariance are

$$\hat{y}(x) = \hat{\beta} + \mathbf{k}_0^T \mathbf{\Psi}^{-1} \left(y^{(k)} - \hat{\beta} \right) \tag{B.6a}$$

$$s^2(x) = \mathbb{k}(x_0, x_0) - \mathbf{k}_0^T \mathbf{\Psi}^{-1} \mathbf{k}_0 + \frac{\left(1 - 1^T \mathbf{\Psi}^{-1} \mathbf{k}_0 \right)^2}{(1^T \mathbf{\Psi}^{-1} 1)} \tag{B.6b}$$

where 1 is the $k \times 1$ column vector of ones. The integrals in equation (B.5) can be reformulated as

$$\mathbb{E}_{\overline{y}_p}(x) = \beta + z^T \mathbf{K}^{-1} \left(y^{(k)} - \beta\right) \tag{B.7a}$$

$$\mathrm{var}(\overline{y}_p) = c + \left[1 - 2z^T \mathbf{\Psi}^{-1} 1\right] \left(1^T \mathbf{\Psi}^{-1} 1\right)^{-1} \tag{B.7b}$$

$$- z^T \mathbf{K}^{-1} \left[\mathbf{I} - 1 \left(1^T \mathbf{\Psi}^{-1} 1\right)^{-1} 1 \mathbf{\Psi}^{-1}\right] z \tag{B.7c}$$

where

$$z_i = \int \Bbbk(x, x^{(i)}) p(x) dx, \qquad i = 1, \ldots, k \tag{B.8a}$$

$$c = \iint \Bbbk(x, x_0) p(x) p(x_0) dx dx_0 \tag{B.8b}$$

In general, combining (B.6) with (B.5) leads to nonanalytic expressions, but if the density $p(x)$ and covariance function (B.4) are Gaussian, analytical results can be obtained. In particular, if $p(x) \sim \mathcal{N}(b, B)$, and $\Bbbk(\cdot, \cdot)$ is specified by equation (B.4),

$$z_i = \theta_0 \left|A^{-1} B + I\right|^{-1/2} \exp\left[-\frac{1}{2}(x^{(i)} - b)^T (A + B)^{-1} (x^{(i)} - b)\right] \tag{B.9a}$$

$$c = \theta_0 \left|2A^{-1} B + I\right|^{-1/2} \tag{B.9b}$$

Figure 97 illustrates a one-dimensional application of BMC to the integral (B.1) with $y(x) = \mathrm{sinc} x$ and $p(x) = \mathcal{N}(0, 1)$ against 4, 8, and 12 samples drawn from $p(x)$. The left column of Figure 97 shows the GP fit to $y(x)$ versus sample size. The right column shows the distribution of $\overline{y}_p$. In general, as more samples are added, the GP fit will improve and the distribution of $\overline{y}_p$ will become more peaked and converge to the exact value.

Figure 98 demonstrates the convergence rates of SMC and BMC versus sample size on two multimodal scalar test functions,

$$y(x) = 2 + \tfrac{1}{20}(x + 4)(x + 2)(x + 1)(x - 1)(x - 3) \tag{B.10}$$

and

$$y(x) = x \left(\sin(10x + 1) + 0.1 \sin 15x\right) \tag{B.11}$$

integrated with respect to $p(x) = \mathcal{N}(-1, 1)$ and $p(x) = \mathcal{N}(-0.5, 0.1)$, respectively. The functions are shown in the first row of plots. The second row of plots shows the $\log_{10}$ of the mean squared error between the exact value of the integral (B.1) (obtained by numerical integration) and the prediction of the method (the mean of the distribution of $\overline{y}_p$ is used for BMC) over 100 repetitions per sample size, i.e., for each sample size k, k samples were drawn from $p(x)$ 100 times and each method was applied to each generated set. As expected, SMC converges at a rate of $1/k$. BMC converges at a rate of approximately $1/k^2$ for both examples. The third row of plots in Figure 98 is a set of box plots depicting the performance spread for each method over 100 repetitions per sample size. Outliers are taken as data points that fall outside of the range [99]

$$[Q_1 - c(Q_3 - Q_1), Q_2 - c(Q_3 - Q_1)] \tag{B.12}$$

with $c = 1.5$ and are represented by crosses. Some interesting properties of BMC can be seen in the second example with function (B.11). In this example, both methods converge to the wrong value because outside A, the function (B.11) oscillates with very large magnitude, so random samples taken in that region add or subtract very large values to the mean. The second observation is that the number of outliers in the second example is substantially higher than the number of SMC outliers. This can be attributed to the failure modes of the MLE approach on function (B.11), which are discussed in detail in Chapters 2 and 3.

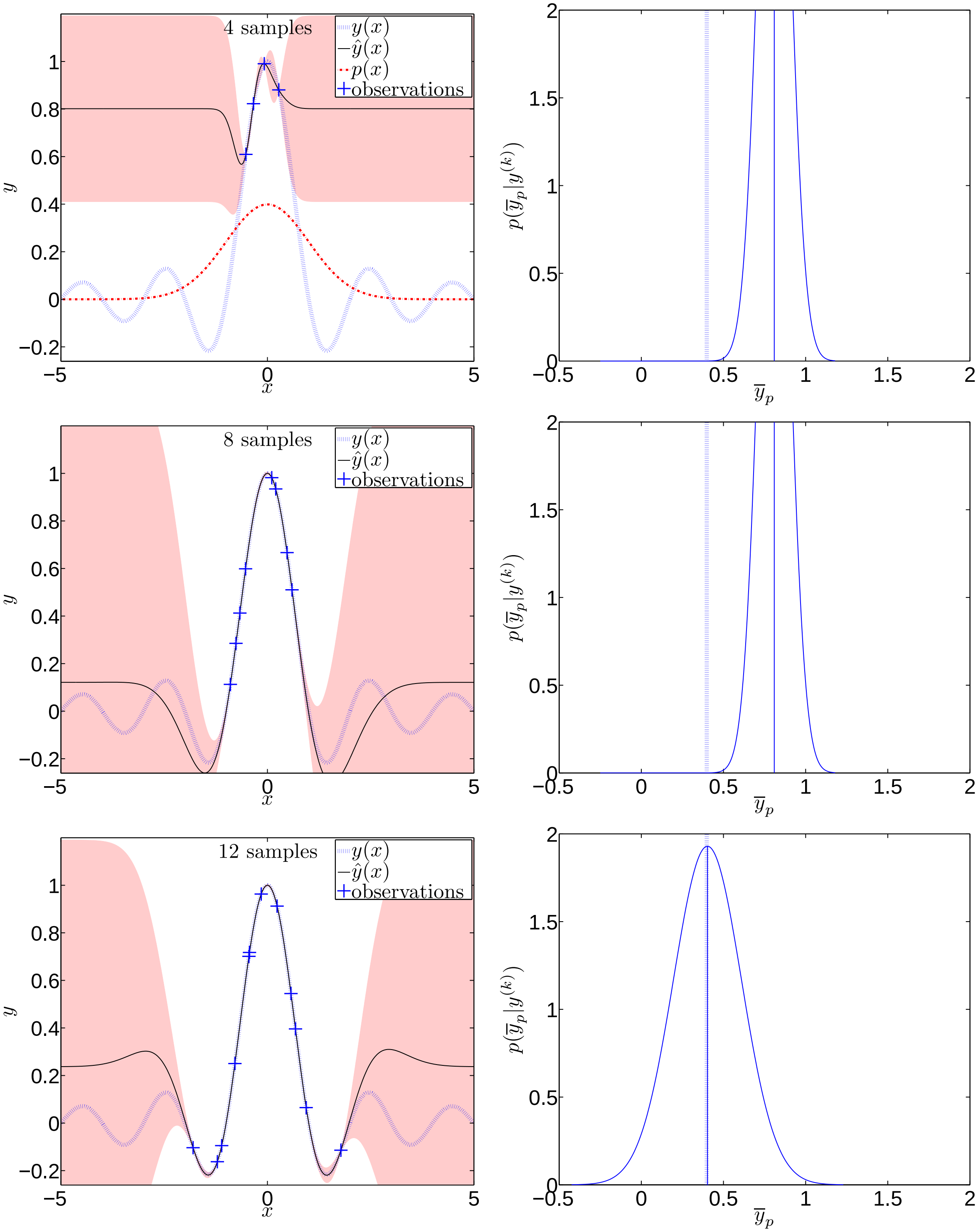


Figure 97: Application BMC to the integral $\overline{y}_p = \int y(x)p(x)dx$ with $y(x) = \text{sinc}x$ and $p(x) = \mathcal{N}(0, 1)$. Left: GP prediction of $y(x)$; observations were drawn from $p(x)$. Right: The distribution of $\overline{y}_p$; true value of $\overline{y}_p$ obtained by numerical integration is shown by the vertical dotted line.

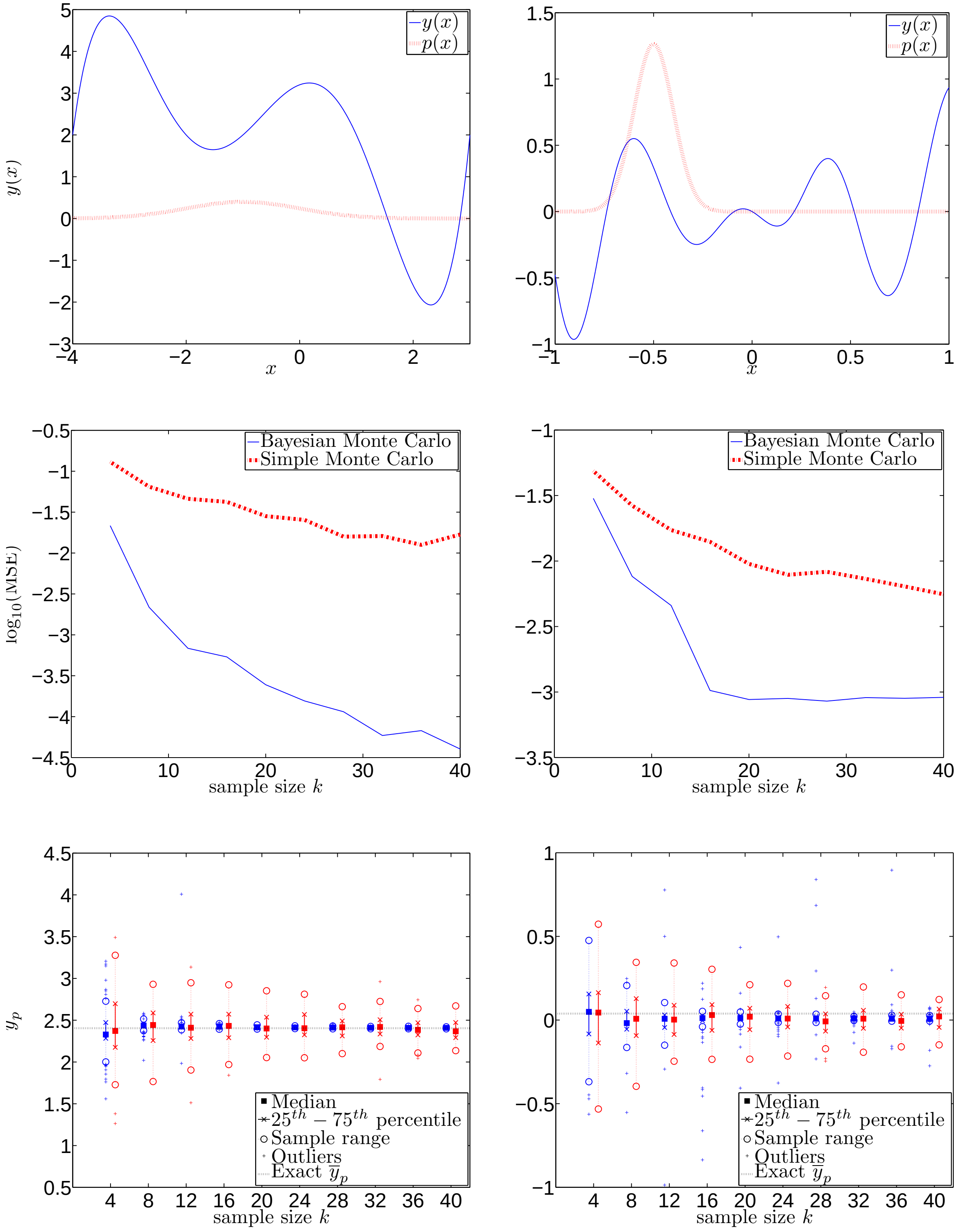


Figure 98: Convergence of BMC and SMC versus sample size for two functions. Top row, left: equation (B.10), $p(x) = \mathcal{N}(-1, 1)$. Top row, right: equation (B.11), $p(x) = \mathcal{N}(-0.5, 0.1)$. Middle row: $\log_{10}$ of the mean squared error between the prediction and the exact value of $\overline{y}_p$ (obtained by numerical integration) over 100 repetitions. Bottom row: box-and-whisker plots of performance versus sample size over 100 repetitions.

REFERENCES

[1] "FAA regulations," http://www.faa.gov/regulations_policies/faa_regulations, 2013, accessed: 26 May 2013.

[2] "Sonic cruiser fact sheet," http://www.boeing.com/news/feature/concept/factsheet.html, 2013, accessed: 9 June 2013.

[3] Abramowitz, M. and Stegun, I. A., (Eds.) *Handbook of mathematical functions*, no. 55 in Applied Mathematics Series, National Bureau of Standards, 1972, tenth ed., URL http://people.math.sfu.ca/~cbm/aands/.

[4] Alexandrov, N., Lewis, R., Gumbert, C., Green, L., and Newman, P., "Optimization with variable-fidelity models applied to wing design," in *Proceedings of the 38th Aerospace Sciences Meeting & Exhibit*, 2000.

[5] Ali, M. M., Khompatraporn, C., and Zabinsky, Z. B., "A numerical evaluation of several stochastic algorithms on selected continuous global optimization test problems," *J. Global Optim.*, vol. 31, pp. 635–672, 2005.

[6] Appleby, D. M., "Probabilities are single-case or nothing," *Opt. Spectrosc.*, vol. 99, no. 3, pp. 447–456, 2005.

[7] Audet, C. and Jr., J. D., "A pattern search filter method for nonlinear programming without derivatives," *SIAM J. Optimiz.*, vol. 14, pp. 980–1010, 2004.

[8] Audet, C., Jr., J. D., Moore, D. W., Booker, A., and Frank, P. D., "A surrogate-model-based method for constrained optimization," in *8th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, AIAA–2000–4891, 2000.

[9] Baritompa, B. and Eligies M.T. Hendrix, E., "On the investigation of stochastic global optimization algorithms," *J. Global Optim.*, vol. 31, pp. 567–578, 2005.

[10] Benassi, R., Bect, J., and Vazquez, E., "Robust gaussian process-based global optimization using a fully bayesian expected improvement criterion," in Coello, C. A. C., (Ed.) *5th international Conference on Learning and Intelligent Optimization (LION 5)*, vol. 6683 of *Lecture Notes in Computer Science*, pp. 176–190, Springer-Verlag, 2011.

[11] Berghen, F. V., *CONDOR: A constraints, nonlinear, derivative-free parallel optimizer for continuous, high computing load, noisy objective functions*, Ph.D. thesis, Universit libre de Bruxelles, 2004.

[12] Bersini, H., Dorigo, M., Langerman, S., Seront, G., and Gambardella, L., “Results of the first international contest on evolutionary optimisation (1st iceo),” in *Proceedings of IEEE International Conference on Evolutionary Computation*, vol. 611–615, 1996.

[13] Betrò, B. and Schoen, F., “A stochastic technique for global optimization,” *Comput. Math. Appl.*, vol. 21, no. 6/7, pp. 127–133, 1991.

[14] Bjorkman, M. and Holmström, K., “Global optimization using the DIRECT algorithm in MATLAB,” *Adv. Model. Optim.*, vol. 1, no. 2, pp. 17–37, 1999.

[15] Bjorkman, M. and Holmström, K., “Global optimization of costly nonconvex functions using radial basis functions,” *Opt. Eng.*, vol. 1, no. 4, pp. 373–397, 2000.

[16] Bombardier, “Commercial aircraft market forecast: 2012–2031,” 2012.

[17] Bongartz, I., Conn, A., Gould, N., and Toint, P. L., “CUTE: Constrained and unconstrained testing environment,” *ACM T. Math. Software*, vol. 21, no. 1, pp. 123–160, 1995.

[18] Booker, A., Jr., J. D., Frank, P., Serafini, D., Torczon, V., and Trosset, M., “A rigorous framework for optimization of expensive functions by surrogates,” *Struct. Optimization*, vol. 17, pp. 1–13, 1999.

[19] Box, G. E. P. and Draper, N. R., *Empirical Model-Building and Response Surfaces*, Wiley, 1987.

[20] Bull, A. D., “Convergence rates of efficient global optimization algorithms,” *Journal of Machine Learning Research*, vol. 12, pp. 2879–2904, 2011.

[21] Calvin, J. and Žilinskas, A., “One-dimensional P-algorithm with convergence rate $O(n^{-3+\delta})$ for smooth functions,” *J. Optimiz. Theory App.*, vol. 106, no. 2, pp. 297–307, 2000.

[22] Calvin, J. M. and Žilinskas, A., *Stochastic and Global Optimization*, vol. 59, chap. One-dimensional global optimization based on statistical models, pp. 49–63, Kluwer Academic, 2002.

[23] Campana, E. F., Liuzzi, G., Lucidi, S., Peri, D., Piccialli, V., and Pinto, A., “New global optimization methods for ship design problems,” *Optim. Eng.*, vol. 10, pp. 533–555, 2009.

[24] Clark, B. J., “Computer program to predict aircraft noise levels,” Tech. Rep. TP-1913, NASA, 1981.

[25] Conn, A., Scheinberg, K., and Toint, P. L., “A derivative free optimization algorithm in practice,” in *Proceedings of 7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, 1998.

[26] Conn, A. R., Scheinberg, K., and Vicente, L. N., *Introduction To Derivative-Free Optimization*, SIAM, 2009.

[27] Cox, D. and John, S., "A statistical method for global optimization," in *Proceedings of the 1992 International Conference on Systems, Man, and Cybernetics*, vol. 2, pp. 1241–1246, 1992.

[28] Cox, S. E., Haftka, R. T., Barker, C. A., Grossman, B., Mason, W. H., and Watson, L. T., "A comparison of global optimization methods for the design of a high-speed civil transport," *J. Global Optim.*, vol. 21, pp. 415–433, 2001.

[29] Dixon, L. and Szego, G., "The global optimization problem: An introduction," in Dixon, L. and Szego, G., (Eds.) *Towards Global Optimization 2*, pp. 1–15, North-Holland, 1978.

[30] Donaldson, B. K., *Analysis of Aircraft Structures: An Introduction*, Cambridge University Press, 2008, second ed.

[31] Drela, M., *Low reynolds number aerodynamics*, vol. 54 of *Lecture Notes in Engineering*, chap. XFOIL: An analysis and design system for low Reynolds number airfoils, Springer-Verlag, 1989, URL `http://web.mit.edu/drela/Public/papers/xfoil_sv.pdf`.

[32] Drela, M. and Youngren, H., "XFOIL 6.94," 2001, URL `http://web.mit.edu/drela/Public/web/xfoil/`.

[33] Feagin, R. C. and Morrison, W. D., "Delta method, an empirical drag buildup technique," Tech. Rep. CR-151971, NASA, 1978.

[34] Fink, D., "A compendium of conjugate priors," 1997.

[35] Finkel, D. E. and Kelley, C. T., "Additive scaling and the DIRECT algorithm," *J. Global Optim.*, vol. 36, no. 4, pp. 597–608, 2006.

[36] Floudas, C. and Pardalos, P., *A Collection of Test Problems for Constrained Global Optimization Algorithms*, no. 455 in Lecture Notes in Computer Science, Springer-Verlag, 1990.

[37] Floudas, C. A., anos M. Pardalos, P., Adjiman, C. S., Esposito, W. R., Gümüs, Z. H., Harding, S. T., Klepeis, J. L., Meyer, C. A., and Schweiger, C. A., *Handbook of Test Problems in Local and Global Optimization*, Kluwer Academic, 1999.

[38] Forrester, A., Keane, A. J., and Bressloff, N. W., "Design and analysis of 'noisy' computer experiments," *AIAA J.*, vol. 44, no. 10, pp. 2331–2339, 2006.

[39] Forrester, A. I., Sóbester, A., and Keane, A. J., *Engineering Design via Surrogate Modelling – A Practical Guide*, John Wiley & Sons, Inc., 2008.

[40] Forrester, A. I. J., *Efficient Global Aerodynamic Optimisation Using Expensive Computational Fluid Dynamics Simulations*, Ph.D. thesis, University of Southampton, 2004.

[41] Forrester, A. I. J., Sóbester, A., and Keane, A. J., "Multi-fidelity optimization via surrogate modelling," *P. Royal. Soc. A. Math.-Phy.*, vol. 463, no. 2088, pp. 3251–3269, 2007.

[42] Gablonsky, J. and Kelley, C., "A locally-biased form of the DIRECT algorithm," *J. Global Optim.*, vol. 21, no. 1, pp. 27–37, 2001.

[43] Geiselhart, K. A., "A technique for integrating engine cycle and aircraft configuration optimization," Tech. Rep. CR-191602, NASA, 1994.

[44] Gutmann, H., "A radial basis function method for global optimization," *J. Global Optim.*, vol. 19, pp. 201–227, 2001.

[45] Handcock, M. S. and Stein, M. L., "A bayesian analysis of kriging," *Technometrics*, vol. 35, no. 4, pp. 403–410, 1993.

[46] Hansen, E., "Global optimization usng interval analysis: The one-dimensional case," *J. Optimiz. Theory App.*, vol. 29, no. 3, pp. 331–344, 1979.

[47] Harris, C. D., "NASA supercritical airfoils: A matrix of family-related airfoils," Tech. Rep. NASA Technical Paper 2969, NASA, Langley Research Center, 1990.

[48] Henkenjohann, N. and Kunert, J., "An efficient sequential optimization approach based on the multivariate expected improvement criterion," *Qual. Eng.*, vol. 19, no. 4, pp. 267–280, 2007.

[49] Hock, W. and Schittkowski, K., *Test Examples for Nonlinear Programming Codes*, Springer-Verlag, 1981.

[50] Holden, C. M. and Keane, A. J., "Visualizastion methodologies in aircraft design," in *10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, AIAA 2004-4449, pp. 1–13, Albany, New York, 2004, 30 August – 1 September.

[51] Jeffreys, H., *Theory of probability*, Clarendon Press, 1983, third ed.

[52] Johnson, F. T., Tinoco, E. N., and Yu, N. J., "Thirty years of development and application of CFD at Boeing Commercial Airplanes, Seattle," *Comput. Fluids*, vol. 34, pp. 115–1151, 2005, also AIAA-2003-3439, June, 2003.

[53] Jones, D., Perttunen, C., and Stuckman, B., "Lipschitzian optimization without the Lipschitz constant," *J. Optimiz. Theory App.*, vol. 79, pp. 157–181, 1993.

[54] Jones, D. R., "A taxonomy of global optimization methods based on response surfaces," *J. Global Optim.*, vol. 21, no. 4, pp. 345–383, 2001.

[55] Jones, D. R., *Encyclopedia of optimization*, chap. Direct global optimization algorithm, pp. 725–735, Springer, 2009, second ed.

[56] Jones, D. R., Schonlau, M., and Welch, W. J., "Efficient global optimization of expensive black-box functions," *J. Global Optim.*, vol. 13, pp. 455–492, 1998.

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

[58] Keynes, J. M., *A treatise on probability*, chap. The principle of indifference, pp. 41–64, MacMillan and Co., 1921.

[59] Khompatraporn, C., Zabinsky, Z. B., and Pintér, J. D., "Comparative assessment of algorithms and software for global optimization," *J. Global Optim.*, vol. 31, pp. 613–633, 2005.

[60] Kirby, M. R., *A methodology for technology identification, evaluation, and selection in conceptual and preliminary aircraft design*, Ph.D. thesis, Georgia Institute of Technology, 2001.

[61] Koullias, S., Balestrini-Robinson, S., and Mavris, D. N., "Surface effect ship sizing and synthesis: A nonlinear programming approach," *11th International Conference on Fast Sea Transportation*, 2011.

[62] Kushner, H., "A new method for locating the maximum point of an arbitrary multipeak curve in the presence of noise," *J. Basic. Eng.*, vol. 86, pp. 97–106, 1964.

[63] Mason, W., Knill, D., Giunta, A., Grossman, B., and Watson, L., "Getting the full benefits of CFD in conceptual design," in *16th AIAA Applied Aerodynamics Conference*, Albuquerque, NM, June 15–18, 1998, also AIAA 98-2513.

[64] Mavris, D. N., Galloway, T., Marx, W., and Garcia, E., *ALCCA: Aircraft life cycle cost analysis Version 6.0*, Aerospace Systems Design Laboratory, Georgia Institute of Technology, 2001.

[65] McCullers, L. A., *FLOPS: Flight Optimization System Release 6.12*, NASA Langley Research Center, 2004.

[66] Messine, F., *Essays and Surveys in Global Optimization*, chap. A deterministic global optimization algorithm for design problems, pp. 267–294, Springer, 2005.

[67] Meyer, C. A., Floudas, C. A., and Neumaier, A., "Global optimization with nonfactorable constraints," *Ind. Eng. Chem. Res.*, vol. 41, pp. 6413–6424, 2002.

[68] Michal, T. R., "Euler technology assessment for preliminary aircraft design - unstructured/structured grid NASTD application for aerodynamic analysis of an advanced fighter/tailless configuration," Tech. Rep. NASA/CR-1998-206947, NASA, 1998.

[69] Mockus, J., *Bayesian Approach to Global Optimization: Theory and Applications*, Kluwer Academic, 1989.

[70] Moré, J. J., Garbow, B. S., and kenneth E. Hillstrom, "Testing unconstrained optimization software," *ACM T. Math. Software*, vol. 7, pp. 17–41, 1981.

[71] Moré, J. J. and Wild, S. M., "Benchmarking derivative-free optimization algorithms," *SIAM J. Optimiz.*, vol. 20, no. 1, pp. 172–191, 2009.

[72] Mozdzanowska, A. and Hansman, R. J., "Growth and operating patterns of regional jets in the united states," *J. Aircraft*, vol. 42, no. 4, pp. 858–864, 2005.

[73] Myers, R. H. and Montgomery, D. C., *Response Surface Methodology: Process and product optimization using designed experiments*, John Wiley & Sons, Inc., 2002.

[74] Nataraj, P. and Arounassalame, M., "Constrained global optimization of multivariate polynomials using bernstein branch and prune algorithm," *J. Global Optim.*, vol. 49, pp. 185–212, 2011.

[75] Nocedal, J. and Wright, S. J., *Numerical Optimization*, Springer, 2006, second ed.

[76] O'Hagan, A., "Bayes-hermite quadrature," *J. Stat. Plan. Infer.*, vol. 29, pp. 245–260, 1991.

[77] O'Hara, J. J., Stump, G. M., Yukish, M. A., Harris, E. N., Hanowski, G. J., and Carty, A., "Advanced visualization techniques for trade space exploration," in *48th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, AIAA 2007-1878, pp. 1–6, Honolulu, Hawaii, 2007, 23 – 26 April.

[78] Osborne, M. A., *Bayesian Gaussian Processes for Sequential Prediction, Optimisation, and Quadrature*, Ph.D. thesis, University of Oxford, 2010.

[79] Osborne, M. A., Garnett, R., and Roberts, S. J., "Gaussian processes for global optimization," *3rd International Conference on Learning and Intelligent Optimization*, 2009.

[80] Parker, J. T., *Naval Postgraduate School Aircraft Synthesis Program (User's Manual)*, Naval Postgraduate School, Monterey, CA, 1991, accessed: 26 May 2013, `http://oai.dtic.mil/oai/oai?verb=getRecord&metadataPrefix=html&identifier=ADA246198`.

[81] Perttunen, C., "A computational geometric approach to feasible region division in constrained global optimization," *IEEE T. Syst. Man. Cyb.*, vol. 1, pp. 585–590, 1991.

[82] Perttunen, C. D. and Stuckman, B. E., "The rank transformation applied to a multiunivariate method of global optimization," *IEEE T. Syst. Man. Cyb.*, vol. 20, no. 5, pp. 1216–1220, 1990.

[83] Pintér, J. D., "Continuous global optimization software: A brief review," 1996, URL `http://plato.la.asu.edu/gom.html`.

[84] Powell, M., "An efficient method for finding the minimum of a function of several variables without calculating derivatives," *Comput. J.*, vol. 7, pp. 155–162, 1964.

[85] Powell, M., "A direct search optimization method that models the objective and constraint functions by linear interpolation," in Gomez, S. and Hennart, J.-P., (Eds.) *Advances in optimization and numerical analysis*, vol. 275, pp. 51–68, Kluwer Academic, 1994.

[86] Powell, M., "On the lagrange functions of quadratic models that are defined by interpolation," *DAMTP 2000/NA10*, 2000.

[87] Powell, M., "UOBYQA: unconstrained optimization by quadratic approximation," *DAMTP 2000/NA14*, 2000.

[88] Powell, M., "The NEWUOA software for unconstrained optimization without derivatives," *DAMTP 2004/NA05 The 40th Workshop on Large Scale Nonlinear Optimization*, 2004.

[89] Powell, M., "The BOBYQA algorithm for bound constrained optimization without derivatives," *DAMTP 2009/NA06*, 2009.

[90] Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P., *Numerical Recipes: The Art of Scientific Computing*, Cambridge University Press, 1986, second ed.

[91] Quttineh, N.-H. and Holmström, K., "Implementation of a one-stage efficient global optimization (EGO) algorithm," *Research Report MDH*, vol. 2, 2009.

[92] Quttineh, N.-H. and Holmström, K., "The influence of experimental designs on the performance of surrogate model based costly global optimization solvers," *Studies in Informatics and Control*, vol. 18, no. 1, pp. 87–95, 2009.

[93] Rasmussen, C. and Ghahramani, Z., "Bayesian monte carlo," in *Advances in Neural Information Processing Systems*, vol. 15, pp. 489–496, MIT Press, 2003.

[94] Rasmussen, C. E. and Williams, C. K. I., *Gaussian Processes for Machine Learning*, The MIT Press, 2006.

[95] Rawson, K. and Tupper, E., *Basic Ship Theory: Volume 1*, Butterworth-Heinemen, 2001, fifth ed.

[96] Raymer, D. P., *Aircraft Design: A Conceptual Approach*, AIAA education series, American Institute of Aeronautics and Astronautics, Reston, Va., 2006, fourth ed., URL `http://www.loc.gov/catdir/toc/ecip068/2006004706.html`.

[97] Regis, R. G. and Shoemaker, C. A., "Constrained global optimization of expensive black box functions using radial basis functions," *J. Global Optim.*, vol. 31, pp. 153–171, 2005.

[98] Regis, R. G. and Shoemaker, C. A., "Improved strategies for radial basis function methods for global optimization," *J. Global Optim.*, vol. 37, pp. 113–135, 2007.

[99] Renze, J., "Outlier," 2012, URL `http://mathworld.wolfram.com/Outlier.html`.

[100] Rizzi, A., "Modeling and simulation aircraft stability and control - the SimSAC project," *Prog. Aerosp. Sci.*, vol. 47, pp. 573–588, 2011.

[101] Robinson, G. and Keane, A., "Concise orthogonal representation of supercritical aerofoils," *J. Aircraft*, vol. 38, no. 3, pp. 580–583, 2001.

[102] Rogers, S. E., Roth, K., Cao, H. V., Slotnick, J. P., Whitlock, M., Nash, S. M., and Baker, M. D., "Computation of viscous flow for a Boeing 777 aircraft in landing configuration," *J. Aircraft*, vol. 38, no. 6, pp. 1060–1068, 2001.

[103] Sacks, J., Welch, W. J., Mitchell, T. J., and Wynn, H. P., "Design and analysis of computer experiments," *Stat. Sci.*, vol. 4, no. 4, pp. 409–435, 1989.

[104] Santner, T. J., Williams, B. J., and Notz, W. I., *The Design and Analysis of Computer Experiments*, Springer, 2003.

[105] Sasena, M., *Flexibility and efficiency enhancements for constraints global design optimization with kriging approximations*, Ph.D. thesis, University of Michigan, 2002.

[106] Sasena, M., Papalambros, P., and Goovaerts, P., "Global optimization of problems with disconnected feasible regions via surrogate modeling," in *9th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, 2002.

[107] Sasena, M. J., Papalambros, P. Y., and Goovaerts, P., "The use of surrogate modeling algorithms to exploit disparities in function computation time within simulation–based optimization," in *Fourth World Congress of Structural and Multidisciplinary Optimization*, 2001.

[108] Schonlau, M., Welch, W., and D.R. Jones, D. R., "Global versus local search in constrained optimization of computer models," 1997.

[109] Sherali, H. D. and Ganesan, V., "A pseudo-global optimization approach with application to the design of containerships," *J. Global Optim.*, vol. 26, pp. 335–360, 2003.

[110] Sobester, A., Leary, S. J., and Keane, A. J., "On the design of optimization strategies based on global response surface approximation models," vol. 33, pp. 31–59, 2005.

[111] Sommer, S. C. and Short, B. J., "Free-flight measurements of turbulent-boundary-layer skin friction in the presence of severe aerodynamic heating at high Mach numbers from 2.8 to 7.0," Tech. Rep. TN-3391, NACA, 1955.

[112] Soms, A. P., "An asymptotic expansion for the tail area of the t-distribution," *J. Am. Stat. Assoc.*, vol. 71, no. 355, pp. 728–730, 1976.

[113] Stuckman, B., "A global search method for optimizing nonlinear systems," in *Proceedings of the IEEE conference on systems, man, and cybernetics*, vol. 1, pp. 965–977, 1989.

[114] Torn, A. and Žilinskas, A., *Global Optimization*, vol. 350 of *Lecture Notes in Computer Science*, Springer-Verlag, 1989.

[115] Tuy, H., *Convex Analysis and Global Optimization*, vol. 22 of *Nonconvex Optimization and Its Applications*, Kluwer Academic, 1998.

[116] van Dam, E., den Hertog, D., Husslage, B., , and Rennen, G., "Space-filling designs," 2009, URL `http://www.spacefillingdesigns.nl/`.

[117] Vanderplaats, G. N., *Numerical Optimization Techniques for Engineering Design*, Garret N. Vanderplaats, 2005, fourth ed.

[118] Vazquez, E. and Bect, J., “Convergence properties of the expected improvement algorithm with fixed mean and covariance functions,” *J. Stat. Plan. Infer.*, vol. 140, no. 11, pp. 3088–3095, 2010.

[119] Villemonteix, J., Vazquez, E., Sidorkiewicz, M., and Walter, E., “Global optimization of expensive-to-evaluate functions: an empirical comparison of two sampling criteria,” *J. Global Optim.*, vol. 43, pp. 373–389, 2009.

[120] Vos, J., Rizzi, A., Darracq, D., and Hirschel, E., “Navier-Stokes solvers in European aircraft design,” *Prog. Aerosp. Sci.*, vol. 38, pp. 60–697, 2002.

[121] Žilinskas, A., “Optimization of one-dimensional multimodal functions,” *J. Roy. Stat. Soc. Ser. C – App.*, vol. 27, no. 3, pp. 367–375, 1978.

[122] Žilinskas, A., “Two algorithms for one-dimensional multimodal minimization,” *Math. Oper. Stat., Ser. Optimization*, vol. 12, no. 1, pp. 53–63, 1981.

[123] Žilinskas, A., “Axiomatic characterization of a global optimization algorithm and investigation of its search strategy,” *Oper. Res. Lett.*, vol. 4, no. 1, pp. 35–39, 1985.

[124] Žilinskas, A., “A review of statistical models for global optimization,” vol. 2, no. 2, pp. 145–153, 1992.

[125] Žilinskas, A. and Žilinskas, J., “P-algorithm based on a simplicial statistical model ofmultimodal functions,” *TOP*, vol. 18, pp. 396–412, 2010.

[126] Wolpert, D. H. and Macready, W. G., “No free lunch theorems for optimization,” in *IEEE Transactions on Evolutionary Computation*, vol. 1, 1997.

Methodology For Global Optimization Of Computationally Expensive Design Problems

Stefanos Koullias

256 Pages

Directed by Dr. Dimitri N. Mavris

The design of unconventional aircraft requires early use of high-fidelity physics-based tools to search the unfamiliar design space for optimum designs. Current methods for incorporating high-fidelity tools into early design phases for the purpose of reducing uncertainty are inadequate due to the severely restricted budgets that are common in early design as well as the unfamiliar design space of advanced aircraft. This motivates the need for a robust and efficient global optimization algorithm.

This research presents a novel surrogate model-based global optimization algorithm to efficiently search challenging design spaces for optimum designs. The algorithm searches the design space by constructing a fully Bayesian Gaussian process model through a set of observations and then using the model to make new observations in promising areas where the global minimum is likely to occur. The algorithm is incorporated into a methodology that reduces failed cases, infeasible designs, and provides large reductions in the objective function values of design problems.

Results on four sets of algebraic test problems are presented and the methodology is applied to an airfoil section design problem and a conceptual aircraft design problem. The method is shown to solve more nonlinearly constrained algebraic test problems than state-of-the-art algorithms and obtains the largest reduction in the takeoff gross weight of a notional 70-passenger regional jet versus competing design methods.