

Uncertainty quantification for flow and transport in porous media

DAVID CREVILLÉN GARCÍA

Thesis submitted to the University of Nottingham
for the degree of Doctor of Philosophy

March 2016

Dedicated to my wife, Bárbara, my son, David and my daughter, Lucía, who have been a constant source of inspiration and support throughout the past years.

Abstract

The major spreading and trapping mechanisms of carbon dioxide in geological media are subject to spatial variability due to heterogeneity of the physical and chemical properties of the medium. For modelling to make a useful contribution to the understanding of carbon dioxide sequestration and its associated risk assessment, the impact of heterogeneity on flow, transport and reaction processes and their uncertainties must be identified, characterised, and its consequences quantified. Complex computer simulation models based on systems of partial differential equations with random inputs are often used to describe the flow of groundwater through this heterogeneous media.

The Monte Carlo method is a widely used and effective approach to quantify uncertainty in such systems of partial differential equations with random inputs. In this method, the relevant parameter values (the properties of the medium) are drawn from their probability distributions, the governing equations are solved, and this is repeated for many such samples. This gives a set of samples of the output variables, from which various statistical quantities of interest, such as mean values, variances, and estimates of cumulative distributions functions, can be calculated.

This thesis investigates two alternatives to Monte Carlo for solving the equations with random inputs; the first of these are techniques developed for improving the computational performance of Monte Carlo, namely methods such as, multi-level Monte Carlo, quasi Monte Carlo, multilevel quasi Monte Carlo. The second alternative is an approach based on Bayesian non parametric modelling, in which we build statistical approximations of the simulator, called emulators. In this latter approach, the relationship between the inputs and outputs of the simulator

is modelled using Gaussian processes that are conditioned on the output of the simulator at carefully chosen inputs, called training points. The emulator is then used as a surrogate for the full simulator in a classical Monte Carlo calculation. The advantage of this approach is that the emulator is much faster than the original simulator, so it is feasible to run it many times in order to perform the desired analysis. The idea behind Gaussian process emulation methodology, is that assuming the simulator is a smoothing varying function of the input parameters, then information can be obtained from a relatively small number of runs of the model. This information can then be used to make inferences about the output of the model given any other input.

Numerical calculations carried out in this thesis have demonstrated the effectiveness of the proposed alternatives to the Monte Carlo method for solving two-dimensional model problems arising in groundwater flow and Carbon capture and storage processes. Multilevel quasi Monte Carlo has been proven to be the more efficient method, in terms of computational resources used, among Monte Carlo, multilevel Monte Carlo and quasi Monte Carlo. Gaussian process emulation has been proven to be a reliable surrogate for these simulators and it has been concluded that the use of Gaussian process emulation is a powerful tool which can be satisfactorily used when the physical processes are modelled through computationally expensive simulators.

Acknowledgements

First and foremost, I would like to specially thank Prof. Henry Power for all the support, patience, valuable discussions and advice, constant supervision and, mainly, for believing in me in difficult times.

I would like to thank Prof. Andrew Cliffe, who sadly passed away last January 2014, for his humanity, respect and invaluable source of inspiration.

I thank my supervisors: Dr. Richard Wilkinson and Dr. Marco Iglesias for their advice and for having driven me to make important strides in establishing myself as a researcher.

I would like to thank Prof. Blas Zamora Parra and Dr. José Luis Vicéns Moltó for their support during the preparation of my MSc thesis and for introducing me to the enjoyable world of the computational fluid dynamics.

I thank Prof. Francisco Balibrea Gallego for his support during my Honors Degree studies and for engaging me in the world of dynamical systems with his brilliant professorial lectures.

Second, I thank my parents, Antonio and Lucía, for their effort and support throughout my academic journey.

Third, I thank The University of Nottingham and mainly The School of Mathematics for its support and for making me enjoy every day during the past three years.

Finally, I thank my friends Andrés López Crevillén and Pascual Gil Hernández for their support during the long nights of this academic journey.

I would like to extend my gratitude to my examiners Prof. Marco Dentz and Dr. Paul Matthews for their constructive advice on how to improve this thesis and for an enjoyable viva.

I would also like to acknowledge the EU Panacea project, FP7, grant agreement 282900 for funding this thesis.

Contents

List of Tables	ix
List of Figures	xi
1 Introduction	1
1.1 Uncertainty analysis of computer models	6
1.2 Classical Monte Carlo, quasi Monte Carlo and multilevel methods for solving partial differential equations with random inputs	8
1.2.1 Monte Carlo simulation methods	8
1.2.2 Standard Monte Carlo simulation	9
1.2.3 Multilevel Monte Carlo simulation	12
1.2.4 Quasi Monte Carlo simulation	16
1.2.5 Multilevel Quasi Monte Carlo simulation	18
1.3 GP emulation of computer models	21
1.3.1 GP regression	22
1.3.2 Building the GP emulator	23
1.3.3 Prediction using the GP emulator	24
1.3.4 An Illustrative one dimensional example	26
1.4 Outline of the thesis	28
2 Gaussian process emulation for uncertainty quantification in ground-water flow models	31
2.1 Generation of the design points	32
2.2 The prior covariance function	34
2.3 The prior mean function	36

2.4	The likelihood function	36
2.5	Training our Gaussian process model	37
2.6	Cross-validation	38
2.7	Travel time of a convected particle in groundwater flow.	39
2.8	Mathematical model	39
2.8.1	Governing equations	40
2.8.2	Domain and boundary conditions	41
2.8.3	Quantity of interest	41
2.8.4	Generation of random permeability fields	45
2.8.5	KL decomposition	46
2.9	Using the GP emulator for making predictions	52
2.9.1	GP emulator test and selection	53
2.10	Uncertainty quantification of the travel time of a convected particle in groundwater flow	59
2.10.1	Using Monte Carlo method to estimate the cumulative dis- tribution function of the travel time	60
2.10.2	Using Gaussian process emulation to predict the cumulative distribution function of the travel time	61
2.11	Gaussian process emulation of two-dimensional fields	65
2.11.1	The singular value decomposition method	65
2.11.2	Reduced rank emulation of pressure fields	66
2.11.3	Building the Gaussian process field emulator	69
2.12	Comparison between direct travel time emulation and travel time obtained from pressure field emulation	71
3	Comparison of GP emulation with other methods for UQ in groundwater flow problems	73
3.1	Comparison among Monte Carlo methods for solving PDEs with random inputs	74
3.1.1	Comparison between classical MC and MLMC	75
3.1.2	Comparison between QMC and MLQMC	77
3.1.3	Comparison between MC and QMC	80
3.1.4	Comparison of MC, QMC, MLMC and MLQMC	82

3.1.5	Comparison of MC and GP emulation	83
4	Application of GP emulation to the convectively-enhanced dissolution model	87
4.1	The convectively-enhanced dissolution process	88
4.2	Mathematical model	89
4.2.1	Governing equations	89
4.2.2	Coordinates system	90
4.2.3	Domain and boundary conditions	91
4.2.4	Streamfunction formulation	92
4.2.5	Non-dimensional formulation	93
4.2.6	Quantity of interest	95
4.2.7	Search of numerical solutions for the C-ED model	97
4.2.7.1	Arclength continuation for finding numerical solutions	99
4.2.8	Solutions for the C-ED problem	101
4.2.9	Impact of the rock heterogeneity on the <i>SF</i>	102
4.3	GP emulator for the C-ED problem	110
4.4	Uncertainty distribution of the <i>SF</i> . Case $Ra=60$	115
4.4.1	Using Gaussian process emulation to predict the cumulative distribution function of the surface flux	117
4.4.2	Uncertainty distribution of the <i>SF</i> . Case $Ra=100$	122
4.5	Gaussian process emulation for multiple solutions problems	124
4.5.1	Gaussian process classification	126
4.5.2	UA of the ECDF of the <i>SF</i> for the C-ED problem	127
4.6	Gaussian process emulation of concentration and streamfunction fields	129
4.7	Building the Gaussian process field emulator for the C-ED problem	132
4.8	Comparison between directly predicted surface flux and estimated surface flux from emulated concentration field	136
5	Conclusions and further work	138
5.1	Computational techniques for UQ	139

5.2 Analysis of the C-ED process in CO ₂ geological storage	140
5.3 Gaussian process emulation for UQ	141
Abbreviations	145
Symbols and notation	147
References	154

List of Tables

3.1	MLMC estimation with bounds of the average travel time according to a given MSE= 0.01. The last number of the first column shows the level at which the code stops.	76
3.2	MLMC estimation with bounds of the average travel time according to a given MSE= 0.0064. The last number of the first column shows the level at which the code stops.	76
3.3	MLMC estimation with bounds of the average travel time according to a given MSE= 0.0025. The last number of the first column shows the level at which the code stops.	76
3.4	MLQMC estimation with bounds of the average travel time according to a given MSE= 0.01. The last number of the first column shows the level at which the code stops.	79
3.5	MLQMC estimation with bounds of the average travel time according to a given MSE= 0.0064. The last number of the first column shows the level at which the code stops.	79
3.6	MLQMC estimation with bounds of the average travel time according to a given MSE= 0.0025. The last number of the first column shows the level at which the code stops.	79
3.7	MC and QMC ε^2 -Cost, obtained from the MLMC and MLQMC simulations respectively, according to the given MSE.	82
3.8	Comparison of the travel time estimations obtained with MC and QMC methods at each level based on 25,000 travel time samples.	82

3.9 Prediction of the average travel time with the GP emulator for levels 2, 3, 4 and 5. A zero-mean and SE covariance functions are used in this model. The training set is formed by 256 points, and the number of KL coefficients retained is 18. The respective 95% confidence intervals for each of the predictions are shown in right column.	85
3.10 Prediction of the average travel time with the GP emulator for levels 2, 3, 4 and 5. A zero-mean and $Matérn_{\frac{3}{2}}$ covariance functions are used in this model. The training set is formed by 256 points, and the number of KL coefficients retained is 18. The respective 95% confidence intervals for each of the predictions are shown in right column.	85
3.11 Prediction of the average travel time with the GP emulator for levels 2, 3, 4 and 5. A zero-mean and RQ covariance functions are used in this model. The training set is formed by 256 points, and the number of KL coefficients retained is 18. The respective 95% confidence intervals for each of the predictions are shown in right column.	85

List of Figures

1.1	Example of two grids M_0 and M_1 for the domain $D = [0, 1] \times [0, 1]$	10
1.2	Two samples of the same random permeability field in two consecutive levels to be used as input in the MLMC method.	13
1.3	Pseudo-random and Sobol sequences based sampling comparison over the unit square.	20
1.4	Gaussian process regression example. (a) Samples drawn from the GP prior distribution with mean $m(x) = \frac{1}{4}x^2$ and covariance $k(x, x') = \exp(-\frac{1}{2}(x - x')^2)$. (b) A training set of 6 points. (c) Samples drawn from the posterior distribution. (d) Differences between prior samples and the updated posterior in the light of the training set.	27
1.5	Emulator approximation to $f(x) = 1 + \frac{3}{2} \sin x + x^2$, and uncertainty region corresponding to 95% confidence intervals for different training sets. We notice how the uncertainty region is negligible around training points and it is reduced when including more points to the training set.	28
2.1	Simulated trajectory of the convected particle initially released at the center of the domain for the problem (2.24) with the boundary conditions given in (3.1).	43
2.2	Typical empirical values of hydraulic conductivity and permeability. (Image extracted from Bear (1972)).	44

2.3 Example of a heterogeneous permeability field (top) generated with the KL decomposition method described in Section 2.8.5. Simulated pressure field (bottom left) for the given permeability from equation (2.35). Simulated trajectory (bottom right) of a convected particle released at the center of the domain. The travel time spent for the particle in reaching the right boundary is 0.59 seconds.	50
2.4 Example of heterogeneous permeability field (top) generated with the KL decomposition method described in Section 2.8.5. Simulated pressure field (bottom left) for the given permeability from equation (2.35). Simulated trajectory (bottom right) of a convected particle released at the center of the domain. The travel time spent for the particle in reaching the right boundary is 1.09 seconds.	51
2.5 Emulation diagram. Above, the set of design points, ξ_i , used to generate the log Gaussian RF, \mathbf{K}_i , with a KL decomposition. These \mathbf{K}_i are then used to compute the corresponding travel times τ_i with the simulator f . These τ_i are then used as observed values in the training set. Below, the set of truncated design points, $\xi_i = (\xi_1, \xi_2, \dots, \xi_D)$, to form the training set along with the observed travel times, τ_i . \hat{f} represents the GP emulator which is able to predict the travel time, $\tau_i^* \in \mathbb{R}$, for a given test case $\xi_i^* \in \mathbb{R}^D$. . .	52
2.6 Predicted values (red) and their 95% bounds (black bars) given by the emulator against observed values (blue) for the LOO-CV from a design of 256 points and using a mean-zero function and a <i>SE</i> covariance function. 4.98% of the observed values out of range. . .	54
2.7 Predicted values (red) and their 95% bounds (black bars) given by the emulator against observed values (blue) for the LOO-CV from a design of 256 points and using a mean-zero function and a <i>Matérn_{3/2}</i> covariance function. 6.64% of the observed values out of range.	55

2.8	Predicted values (red) and their 95% bounds (black bars) given by the emulator against observed values (blue) for the LOO-CV from a design of 256 points and using a mean-zero function and a $\text{Matérn}_{\frac{5}{2}}$ covariance function. 6.25% of the observed values out of range.	56
2.9	Predicted values (red) and their 95% bounds (black bars) given by the emulator against observed values (blue) for the LOO-CV from a design of 256 points and using a mean-zero function and a RQ covariance function. 6.25% of the observed values out of range.	56
2.10	Different scores for a GP emulator built with a mean-zero function, SE covariance function and 256 design points. The x-axis represents the number of KL coefficients retained for each score.	57
2.11	Observed and predicted travel times for a design of 256 points, mean-zero function, and $\text{Matérn}_{\frac{3}{2}}$ covariance function. The solid line shows <i>observed</i> $\tau = \text{predicted}$ τ	58
2.12	Relative error curve between observed and predicted surface fluxes for 256 different designs using SE covariance function. The curve shows a smooth decreasing tendency although after around 64 design points retained there is no much improvement in the relative error meanwhile it becomes more computationally expensive.	58
2.13	Monte Carlo ECDF (black line) based on 50000 travel times. The dashed lines show the 95% uncertainty bounds.	61
2.14	GP Posterior ECDFs samples, F_i , approximating the MC estimation of the CDF of the travel time of a convected particle in groundwater flow. The GP emulator prior specifications are: mean-zero function, SE covariance function, 256 design points and 16 KL coefficients retained.	63
2.15	GP uncertainty analysis of the CDF of the travel time of a convected particle in groundwater flow. The GP emulator prior specifications are: mean-zero function, SE covariance function, 256 design points and 16 KL coefficients retained.	64

2.16 True pressure field, P_1 , and corresponding reduced rank approximation, \tilde{P}_1 . Number of eigenvectors used is $k = 12$. The RE between P_1 and \tilde{P}_1 is 0.00042.	67
2.17 True pressure field, P_2 , and corresponding reduced rank approximation, \tilde{P}_2 . Number of eigenvectors used is $k = 16$. The RE between P_2 and \tilde{P}_2 is 0.00581.	68
2.18 RE (Y axis) between each of the true pressure fields and their corresponding reduced rank approximations along the number of eigenvectors retained (X axis).	68
2.19 Observed pressure field, P , and corresponding reduced rank approximation, \tilde{P} , and predicted pressure field, P^* . Number of eigenvectors used is $k = 40$. Number of KL coefficients for prediction is 16. The L^2 -norm relative error between P and \tilde{P} is 0.00042 and the L^2 -norm relative error between P and P^* is 0.01594.	70
2.20 Observed τ vs directly predicted SF and Observed τ vs τ from predicted pressure field plots obtained from the simulator. The design was formed by 256 points. A zero-mean and SE covariance function were used as prior GP model specifications and the number of KL coefficients used for prediction was 16. The solid line shows <i>observed</i> $\tau =$ <i>predicted</i> τ	72
3.1 Performance plots for the expectation in the MLMC method. The plots show the numerical verification of the asymptotic behaviour of the expectation of T and the convergence of $\mathbb{E}[Y_\ell]$	77
3.2 Performance plots for the variance in the MLMC method. The plots show the numerical verification of the asymptotic behaviour of the variance of T and the convergence of $\mathbb{V}[Y_\ell]$	78
3.3 Performance plots for the expectation in the MLQMC method. The plots show the numerical verification of the asymptotic behaviour of the expectation of T and the convergence of $\mathbb{E}[Y_\ell]$	80
3.4 Performance plots for the variance in the MLQMC method. The plots show the numerical verification of the asymptotic behaviour of the variance of T and the convergence of $\mathbb{V}[Y_\ell]$	81

3.5	Analysis of the convergence of MC and QMC methods for the average travel time at levels 3, 4 and 5. The convergence is calculated over a sample of 25,000 travel times.	83
3.6	ε^2 -Cost for MC, QMC, MLMC and MLQMC methods for MSE: 0.01, 0.0064 and 0.0025.	84
3.7	MC converge analysis (blue line) based of 10^5 travel time samples at each level (1-5). RMSE tolerances (black vertical bars) for levels 3-5: RMSE=0.1, RMSE= 0.08 and RMSE=0.05 respectively. T_{GP}^{SE} , $T_{GP}^{3/2}$ and T_{GP}^{RQ} are the predicted average travel time for each of the GP emulators.	86
4.1	Two dimensional domain with horizontal length L and vertical depth $2H$, i.e., $D = [0, L] \times [-H, H]$. Gravity \mathbf{g} acts downwards along with the Z-axis.	91
4.2	Bifurcation diagram with respect to Ra . Blue lines shows steady-state solutions projected on the streamfunction value at the center of the domain, C_c . The other model parameters are $K = 1$, $Da=0.1$ and $\beta_T = \beta_L = 0$	99
4.3	Bifurcation diagram with respect to Ra and $\varepsilon > 0$. Blue lines shows steady-state solutions projected on the concentration value at the center of the domain, C_c . The other parameters of the model are $Da=0.1$, $\beta_L = \pi/2$ and $\beta_T=\beta_L/10$	100
4.4	Illustrative representation of the bifurcation scenery against the Ra for a given heterogeneous permeability field. The red circles represents three different numerical solutions of the C-ED problem with parameters: $Ra=60$, $Da=0.1$, $\beta_L = \pi/2$ and $\beta_T=\beta_L/10$. The blue horizontal line represents the no-flow solutions branch.	101
4.5	Contours of the concentration (left) and streamfunction (right) for a given permeability field (top) for the C-ED problem with parameters: $Ra=60$, $Da=0.1$, $\beta_L = \pi/2$ and $\beta_T=\beta_L/10$	104
4.6	Contours of the concentration (left) and streamfunction (right) for a given permeability field (top) for the C-ED problem with parameters: $Ra=60$, $Da=0.1$, $\beta_L = \pi/2$ and $\beta_T=\beta_L/10$	105

4.12 Predicted values (red) and their 95% bounds (black bars) given by the emulator against observed values (blue) for the LOO-CV from a design of 174 points and using a mean-zero function and a $\text{Matérn}_{\frac{3}{2}}$ covariance function. 4.02% of the observed values out off range.	112
4.13 Different scores for a GP emulator built with a mean-zero function, $\text{Matérn}_{\frac{3}{2}}$ covariance function and 174 design points. The parameters of the C-ED simulator are: $Ra = 60$, $Da = 0.1$, $\beta_L = \pi/2$, and $\beta_T = \beta_L/10$. The x-axis represents the number of KL coefficients retained for each score.	113
4.14 Observed and predicted surface fluxes for a design of 174 points, mean-zero function, and $\text{Matérn}_{\frac{3}{2}}$ covariance function. The solid line shows <i>observed SF</i> = <i>predicted SF</i>	114
4.15 Relative error between 1000 observed and predicted surface fluxes against the number of design points retained using the $\text{Matérn}_{\frac{3}{2}}$ covariance function. The curve shows a decreasing tendency and it shows that after around 32 design points retained there is no variation in the relative error.	115
4.16 Monte Carlo ECDF (black line) based on 1000 surface fluxes for the case $Ra=60$. The dashed lines show the 95% uncertainty bounds.	116
4.17 GP Posterior ECDFs samples, F_i , approximating the MC estimation of the CDF of the travel time of a convected particle in groundwater flow. The GP emulator prior specifications are: mean-zero function, SE covariance function, 174 design points and 14 KL coefficients retained.	117
4.18 GP uncertainty analysis of the CDF of the travel time of a convected particle in groundwater flow. The GP emulator prior specifications are: mean-zero function, SE covariance function, 174 design points and 14 KL coefficients retained.	118

4.19 Predicted values (red) and their 95% bounds (black bars) given by the emulator against observed values (blue) for the LOO-CV from a design of 192 points and using a mean-zero function and a $Matérn_{\frac{3}{2}}$ covariance function. 4.06% of the observed values out off range.	119
4.20 Different scores for a GP emulator built with a mean-zero function, RQ covariance function and 192 design points. The parameters of the C-ED simulator are: $Ra = 100$, $Da = 0.1$, $\beta_L = \pi/2$, and $\beta_T = \beta_L/10$. The x-axis represents the number of KL coefficients retained for each score.	120
4.21 Observed and predicted surface fluxes for a design of 174 points, mean-zero function, and $Matérn_{\frac{3}{2}}$ covariance function. The solid line shows <i>observed SF</i> = <i>predicted SF</i>	121
4.22 Relative error between 1000 observed and predicted surface fluxes against the number of design points retained using the RQ covariance function. The curve shows a decreasing tendency and it shows that after around 20 design points retained there is no variation in the relative error.	121
4.23 Monte Carlo ECDF (blue line) based on 1000 samples for the C-ED simulator. The dotted lines shows uncertainty bounds (95% confidence intervals).	122
4.24 Uncertainty analysis for ECDF when considering 1 KL coefficients (a) and (b), and 8 KL coefficients (c) and (d).	123
4.25 Algorithm followed for predicting the <i>SF</i> for any input given by using a Gaussian process classifier. $U \sim \mathcal{U}(0, 1)$ is an uniform random variable between 0 and 1 which gives randomness to the classification process.	125

4.26 Difference between posterior samples of predicted ECDFs (green) and true <i>SF</i> ECDF (black) based on 1000 samples with and without using GP classification. The number of design points were 256 and the number of KL coefficients were 18. The parameters for the E-CD problem were chosen to be $Ra=100$, $Da=0.1$, $\beta_L = \pi/2$ and $\beta_T = \beta_L/10$. Priors mean-zero, SE covariance, <i>Erf</i> likelihood functions and EP method for inference were chosen for the GP classifier. Priors mean-zero, RQ covariance, Gaussian likelihood functions and exact inference method were chosen for GP regression model.	127
4.27 UA of predicted <i>SF</i> ECDFs based on 1000 samples using GP classification. True ECDF (black), predicted ECDF (red), 2.5th and 97.5th percentiles (dashed magenta). The number of design points were 256 and the number of KL coefficients were 18. The parameters for the E-CD problem were chosen to be $Ra=100$, $Da=0.1$, $\beta_L = \pi/2$ and $\beta_T = \beta_L/10$. Priors mean-zero, SE covariance, <i>Erf</i> likelihood functions and EP method for inference were chosen for the GP classifier. Priors mean-zero, RQ covariance, Gaussian likelihood functions and exact inference method were chosen for GP regression model.	128
4.28 True concentration field, C_1 , and corresponding reduced rank approximation, \tilde{C}_1 . Number of eigenvectors used is $k = 15$. The L^2 -norm relative error between C_1 and \tilde{C}_1 is 0.0014.	130
4.29 True concentration field, C_2 , and corresponding reduced rank approximation, \tilde{C}_2 . Number of eigenvectors used is $k = 15$. The L^2 -norm relative error between C_2 and \tilde{C}_2 is 0.0013.	130
4.30 True concentration field, P_1 , and corresponding reduced rank approximation, \tilde{P}_1 . Number of eigenvectors used is $k = 15$. The L^2 -norm relative error between P_1 and \tilde{P}_1 is 0.0097.	131
4.31 True concentration field, P_2 , and corresponding reduced rank approximation, \tilde{P}_2 . Number of eigenvectors used is $k = 15$. The L^2 -norm relative error between P_2 and \tilde{P}_2 is 0.0037.	131

4.32 Observed concentration field, C , and corresponding reduced rank approximation, \tilde{C} , and predicted concentration field, C^* . Number of eigenvectors used is $k = 40$. Number of KL coefficients for prediction is 16. The L^2 -norm relative error between C and \tilde{C} is 0.0031 and the L^2 -norm relative error between C and C^* is 0.0370.	133
4.33 Observed streamfunction field, Ψ , and corresponding reduced rank approximation, $\tilde{\Psi}$, and predicted streamfunction field, Ψ^* . Number of eigenvectors used is $k = 40$. Number of KL coefficients for prediction is 16. The L^2 -norm relative error between Ψ and $\tilde{\Psi}$ is 0.0111 and the L^2 -norm relative error between Ψ and Ψ^* is 0.1887.	134
4.34 Observed, reduced rank approximation and predicted streamfunction fields. Number of eigenvectors used is $k = 40$. Number of KL coefficients for prediction is 16. The L^2 -norm relative error between the observed field and reduced rank approximation is 0.0031 and the L^2 -norm relative error between the observed field and the predicted is 0.1521.	135
4.35 RE (Y axis) between each of the observed concentration field and its corresponding reduced rank approximation along the number of eigenvectors used (X axis).	136
4.36 Observed SF vs directly predicted SF and Observed SF vs SF from predicted concentration field plots obtained from the simulator for the E-CD problem for $Ra=100$, $Da=0.1$, $\beta_L = \pi/2$ and $\beta_T = \beta_L/10$. The design was formed by 123 points (one cell solutions only). A zero-mean and RQ covariance function were used as prior GP model specifications and the number of KL coefficients used for prediction was 16. The solid line shows $observed SF = predicted SF$.	137

Chapter 1

Introduction

The quality of the Earth's water resources has been declining since the beginning of the industrial revolution. Water resources are composed of ground and surface water, and keeping these clean of pollution is of vital importance. Contamination of groundwater can occur in numerous ways, including accidental spills, spent nuclear fuel repositories, and from industrial processes such as carbon capture and storage (CCS) and fracking. The use of aquifers as a potable water supply and the potential adverse effects that contamination can pose to users, necessitates the development of methods for predicting the transport of contaminants in aquifers.

Scientific contributions to this problem include the use of mathematical and computational tools to predict the physical behaviour of the fluids or solutes that occur deeply underground. Solute transport in groundwater occurs mainly under convection and dispersion effects. Path lines of solute particles and their arrival times at receptors (e.g., canals or rivers), broadly known as *particle tracking*, are important in practice, for instance, the leakage of radionuclides which are transported through groundwater flow and which has direct impact on water resources. Convection models by themselves cannot be used to compute solute particles in groundwater flow because they do not consider the effect of mixing by dispersion. On the other hand, convection models represent a valuable intermediate step between groundwater flow models and convection-dispersion-reaction solute transport models. In the first part of this thesis we analyse this phenomenon by applying some of the current numerical techniques for solving the mathematical models used to predict this physical process. The fact that we deal with a small-

scale problem provide us with an excellent opportunity for deeply analysing the methodologies proposed, and thus, help us to succeed when attempting to solve real large-scale problems occurring in CCS technology.

Climate change concerns governments and people around the world (IPCC, AR5). There is a drive for all countries to reduce their carbon dioxide (CO_2) emissions. CO_2 emissions arising from the utilization of fossil fuels are expected to provoke climate change over the next century (IPCC, AR5), and thus, research has been done on methods to reduce greenhouse gas emissions. CCS is an attractive technology because it provides the chance of maintaining access to fossil fuel energy, while cutting CO_2 emissions into the atmosphere. Therefore, CCS is presented as a global warming mitigation strategy and will be a practical method if the sequestered CO_2 can be retained and trapped safely underground. CO_2 can be trapped in saline geological formations by three mechanisms: firstly, it can be trapped as a residual phase (residual phase trapping). Secondly, it can be dissolved into the formations brine (solubility trapping). This enhances the acidity of the brine and increases the solubility of many minerals composed of the host rock matrix. And thirdly, CO_2 may dissolve and dissociate into ionic species and react with minerals in the geologic formation, leading to the precipitation of secondary carbonate minerals (mineral trapping). In this thesis we are interested in investigating the latter mechanism.

The major spreading and trapping mechanisms of carbon dioxide in geological media are subject to spatial variability due to heterogeneity of the physical and chemical properties of the medium. Heterogeneity acts on the multi-phase flow properties of the carbon dioxide-brine system and can lead to trapping of brine behind the carbon dioxide phase and increased spread of the CO_2 -brine interface (Bolster et al., 2009). These heterogeneity-induced processes increase the CO_2 -brine contact area and thus can increase the dissolution efficiency of CO_2 . The mixing of the resulting denser carbon dioxide rich water and the reservoir water is due to diffusion and the interaction with spatial heterogeneity and buoyancy effects. The efficiency of chemical reactions due to the mass transfer limitations and interaction with the medium is again subject to spatial heterogeneity in the physical and chemical medium properties (Dentz et al., 2011a). While heterogeneity can lead to increased spreading and mixing of waters with

different chemical compositions, chemical reaction rates for heterogeneous media can be much smaller than the ones obtained in a homogeneous laboratory settings (Dentz et al., 2011b). This behavior can be traced back to mass transfer limitations as a consequence of spatial variability. These effects, along with the influence of capillarity are in general disregarded in large-scale reservoir models. The impact of these heterogeneity and capillarity related processes may be disregarded in petroleum applications, but is critical for assessing the long-term fate of the geological storage of CO₂.

Quantifying the reaction and transport behavior in effective large scale flow, reaction and transport equations is a major challenge. Spatio-temporal fluctuations induce not only scale effects in the major spreading and trapping processes, but cause (due to limited knowledge about the precise medium structure and driving forces) uncertainty about the predicted large scale behavior. Thus, actual predictability of the relevant processes on the large scale requires quantification of the mean behavior and quantification of the fluctuations about it. There are two main approaches to quantifying uncertainty that can be categorised as deterministic and probabilistic. In the deterministic approach, the best possible model is constructed and the impact of uncertainties is considered by running variants of the model. The main difficulty is in deciding which variants to study and making sure that a sufficiently wide range of alternatives have been considered. In the probabilistic approach, the uncertainties are characterised by random variables and the results are presented as the stochastic laws for the outputs. There are a number of difficulties with this approach. Firstly, determining the stochastic laws for the input random variables may be difficult especially if they are correlated. Secondly, there may be a very large number of random input variables. This is particularly so if some of the inputs are spatially distributed, in which case there will essentially be an infinite number of input random variables. Of course, these random fields will be approximated by a finite number of such variables, but this number may be large. Thirdly, this approach is very computationally demanding since many simulations of the basic model have to be run with different parameter values. Finally, it is difficult to treat conceptual model uncertainty in this way because it is not clear how to assign a probability measure to the set of all models (Bannör and Scherer, 2014).

The Monte Carlo (MC) method is a widely used and effective approach to solving systems of partial differential equations with random inputs. In this method the relevant parameter values are drawn from their probability distributions and the governing equations are solved for many such samples. This gives a set of samples of the output variables, from which various statistical quantities of interest, such as mean values, variances, estimates of cumulative distributions functions, can be calculated. The implementation of the classical MC method is straightforward, and it can be applied to any type of problem including non-linear problems, it does not suffer from the so called “curse of dimensionality” (Hadley, 2013) and it is possible to compute an estimate of the error as part of the solution process. The main difficulty with the method is its slow rate of convergence: the error decreases as the inverse of the square root of the number of samples (Cliffe et al., 2011). An alternative to MC methods is provided by probability density function (PDF) approaches (Tartakovsky et al. (2007) and Dentz and Tartakovsky (2010)). This approach derives dynamical equations for the PDFs of the state variables (pressure, concentration, saturation, reaction rates) based on the stochastic partial differential equations governing the phenomenon under consideration (flow, transport, reaction) and allow for a map of the uncertainty of the medium heterogeneity (physical and chemical) and the driving forces on the uncertainty of the system state.

Current multi-phase flow and reactive transport models do not take into account the impact of heterogeneity on front spreading and mass transfer between high and low permeability zones of the heterogeneous medium and the impact of physical and chemical heterogeneity on chemical reactions. To date, the quantification of uncertainty arising from heterogeneities in rock properties and temporal fluctuations in modelling large scale of CO₂ sequestration has not received much attention. The few calculations that have been done have used classical MC methods and have required enormous computational resources. The work proposed in this thesis extends the state of the art by applying the latest developments in the solution of equations with random inputs to the problem of uncertainty quantification (UQ), first in a simple convection model, and second in a large scale CO₂ sequestration model.

The reason UQ becomes important in the analysis of computer models lies

in the fact that some uncertainties may have limited impact on the outcome of performance assessment, and if this can be established, then it is not necessary to explicitly deal with them. Those sources of uncertainty that cannot be treated in this way must be dealt with explicitly. This is a challenging problem. For the large-scale, time-dependent simulations that must be carried out as part of an investigation of a storage site for CO₂ classical MC simulation will be impractical unless considerable computing resources are available. Even if such resources are available, they could be better deployed if more efficient methods are available to solve the equations with random inputs, such as investigating alternative conceptual models and better models of the basic physics.

The work proposed here aims to investigate alternative methods to MC for solving the equations with random inputs. The main source of uncertainty that will be considered is the physico-chemical heterogeneity of the rock formation. Suitable stochastic models for these heterogeneities will be analysed in Chapter 2. Such models have an infinite number of stochastic degrees of freedom and will be approximated by a finite number of degrees of freedom using Karhunen-Loéve (KL) decompositions. To solve the resulting problem two techniques will be investigated. The first methodology consists in the implementation of some of the numerical methods currently used to increase the rate of convergence of MC, namely, multilevel Monte Carlo (MLMC), quasi Monte Carlo (QMC) and multilevel quasi Monte Carlo (MLQMC). These methods have been proven (see e.g., Cliffe et al. (2011) and Giles (2008)) to reduce significantly the asymptotic cost of solving the stochastic problem with the standard MC method. In many applications, the quantity of interest is the expected value of a functional of the solution of the model. Multilevel methods exploits the linearity of expectation, by expressing the quantity of interest on the finest spatial grid in terms of the same quantity on a relatively coarse grid and correction terms. The dramatic reduction in cost associated with the MLMC method over standard MC is due to the fact that most of the uncertainty can be captured on the coarse grids and so the number of realisations needed on the finest grid is greatly reduced. The second approach will be to use statistical approximations of the simulator called emulators. In this approach the relationship between the inputs and outputs of the simulator is modelled as a Gaussian process (GP) that is conditioned on the

output of the simulator for appropriately chosen inputs called training points. The emulator is then run as a surrogate for the full simulator in a classical MC calculation. The advantage of this approach is that the emulator is much faster than the original simulator, so it is feasible to run it many times to achieve the accuracy required.

So far, we have broadly introduced the physical problems and the current methodologies used by scientists to tackle those problems. We have talked about the need to analyse the uncertainties that occur in computer models, but we have not made clear what uncertainty analysis (UA) is.

1.1 Uncertainty analysis of computer models

Throughout this thesis we will call *the simulator* the combination of the system of PDEs used to model a physical phenomenon and the numerical method used to obtain a solution of the given mathematical problem. Henceforth, such a simulator can be regarded as a mathematical function $f(\cdot)$, that takes a vector \mathbf{x} of *inputs* and produces an *output* scalar¹ $y = f(\mathbf{x})$. The outputs of a simulator are a prediction of the real-world phenomena that are simulated by the model, but as such will inevitably be imperfect. There may be many uncertainties when using a computer model to determine the outcome of physical processes. Kennedy and O'Hagan (2001) give a detailed list of these uncertainties and Stone (2011) summarises these uncertainties as follows:

1. There may be uncertainty about the values of the inputs of the computer code. These inputs can be thought of as unknown parameters of the model, and the uncertainty about them is therefore called parameter uncertainty.
2. As the mathematical model may be a simplification of the process, the model will inadequately predict the value of the true process, even if the inputs are known. This is known as structural uncertainty, and is the

¹We will firstly consider the simple case where a simulator output is a scalar. Later on this thesis we will extend this concept to two dimensional field outputs (see 2.11.2).

difference between the true process and the code output at the best values of the input.

3. The model predicts the process under conditions specified by the inputs. However, the process itself may not give the same value under repeated conditions. This residual variability is due to conditions that are not recognised in the mathematical model on which the computer model is based, i.e. structural uncertainty.
4. If any observations of the process are used to calibrate the code, these may include errors. The observation errors add to the uncertainty in the model.
5. The output of the code can also be uncertain. Even though it is a mathematical function of the inputs, it may not be practical to know the output of the code for any set of inputs if the code is complex and takes a long time to run. However, if it is only required to know the output for a small number of inputs code uncertainty would not be a problem.

To above list, we can add the error arisen from the discretization of the physical domain used by the simulator to produce the output, this is sometimes known as *bias*.

Various types of analysis have been used to address some of these uncertainties, for example, uncertainty analysis, sensitivity analysis, calibration and validation. Stone (2011) defines these terms as follows: UA as the process of quantifying the uncertainties in that output due to uncertainties in the inputs. Sensitivity analysis examines how the code output varies in response to changes in inputs, particularly finding out which inputs have the most impact on the output. Calibration relates to changing the parameters of the model, so that the code output fits the observed data, in the sense that the difference between the observed outcome and the model output is small. Validation assesses how well the code predicts reality. In this thesis we are interested in analysing the output of groundwater flow models due to the uncertainties in the inputs. We therefore concentrate on uncertainty analysis.

In the following sections of this introduction we describe, in a more detailed manner, the state of the art of the computational tools that will be used in the

following chapters of this thesis to perform uncertainty quantification for flow and transport in the porous media models proposed.

1.2 Classical Monte Carlo, quasi Monte Carlo and multilevel methods for solving partial differential equations with random inputs

There is a broad family of simulation methods which extend classical Monte Carlo methods. These methods can be used to quantify uncertainty in ground-water flow models arising from solving numerically elliptic partial differential equations with random coefficients.

In this section we give a brief description of the different approaches available for solving PDEs with random coefficients. For a further description of the MC and MLMC methods see Cliffe et al. (2011) and Giles (2008); Graham et al. (2011) and Giles and Waterhouse (2009) for QMC and MLQMC.

1.2.1 Monte Carlo simulation methods

Let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space, where Ω denotes the set of all possible outcomes for a given experiment and it is called *the sample space*, \mathcal{F} is a set of events where each event is a set containing zero or more outcomes, and \mathbb{P} is the assignment of probabilities to the events; that is, a function \mathbb{P} from events to probabilities. Let \mathbf{X}_M be a random vector that takes values in \mathbb{R}^M and let $T_M = f(\mathbf{X}_M)$ be some linear or nonlinear functional¹ of \mathbf{X}_M .

We assume that the expected value $\mathbb{E}[T_M] \rightarrow \mathbb{E}[T]$, for some (inaccessible) random variable $T : \Omega \rightarrow \mathbb{R}$, as $M \rightarrow \infty$, and that (in mean) the order of convergence is $\alpha > 0$ (see Cliffe et al. (2011) and Giles (2008)), i.e.,

$$|\mathbb{E}[T_M - T]| \leq M^{-\alpha}. \quad (1.1)$$

¹Note that this is what we called earlier *the simulator*.

We are interested in estimating $\mathbb{E}[T]$. Thus, given $M \in \mathbb{N}$ sufficiently large, we compute approximations (or estimators) \hat{T}_M of $\mathbb{E}[T_M]$ and quantify the accuracy of our approximations via the root mean square error (RMSE)

$$e(\hat{T}_M) := \left(\mathbb{E}[(\hat{T}_M - \mathbb{E}[T])^2] \right)^{\frac{1}{2}}. \quad (1.2)$$

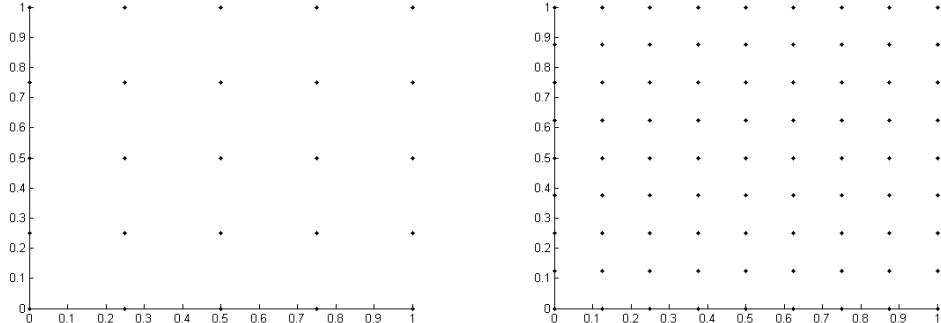
In terms of solving systems of PDEs, a computer model needs to retain all the important features of the physical domain (a continuum medium) of the problem and reduce them into a simplified form, called the computational domain (a discrete set of points). Throughout this thesis we will call *grid* the structured distribution of points, called *nodes*, that form the computational domain used by the computer model to solve the equations, and M will denote the number of nodes which form the corresponding grid. According to this, given two grids M_i and M_j , with $i, j \in \mathbb{N}$, and $i < j$, we will say that M_i is a *subgrid* of M_j , and we will write $M_i < M_j$, if all the nodes contained in M_i are also contained in M_j . We will then say that M_i is *coarser* than M_j and conversely that M_j is *finer* than M_i . For solving efficiently systems of PDEs, choosing M sufficiently large corresponds to choosing a fine enough grid that guarantees that the computer model is providing an accurate approximation of the true solution of the problem. Figure 1.1 shows an example of two grids for the same physical domain used by a computer model.

The random variable T is, in this case, a functional of the solution of our PDE system, and T_M will be the same functional of the discretised solution. We will define, and denote by, \mathcal{C}_ε , the computational ε -cost used to achieve a RMSE of $e(\hat{T}_M) \leq \varepsilon$. This ε -cost is quantified by the number of floating point operations that are needed to achieve a RMSE of $e(\hat{T}_M) \leq \varepsilon$.

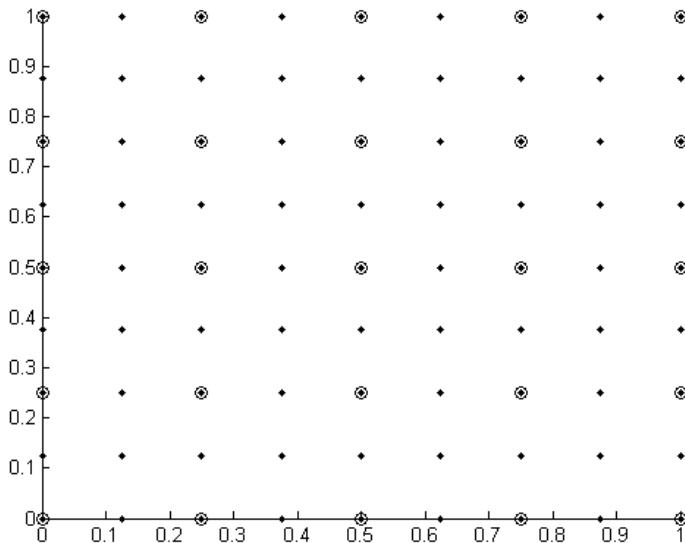
1.2.2 Standard Monte Carlo simulation

We define the standard MC estimator for estimating $\mathbb{E}(T_M)$ as follows,

$$\hat{T}_{M,N}^{MC} := \frac{1}{N} \sum_{i=1}^N T_M^{(i)}, \quad (1.3)$$



(a) Grid with 25 nodes, M_0 , for the domain D . (b) Grid with 81 nodes, M_1 , for the domain D .



(c) Two grids, M_0 (circles) and M_1 (dots), for the same domain D . M_0 is considered a subgrid of M_1 , i.e., $M_0 < M_1$.

Figure 1.1: Example of two grids M_0 and M_1 for the domain $D = [0, 1] \times [0, 1]$.

where $T_M^{(i)}$ is the i th sample of T_M and N independent samples are computed in total. Note that $\mathbb{E}[\hat{T}_{M,N}^{MC}] = \mathbb{E}[T_M]$, i.e., $\hat{T}_{M,N}^{MC}$ is an unbiased estimator of $\mathbb{E}[T_M]$. We assume that the cost to compute one sample $T_M^{(i)}$ of T_M is

$$\mathcal{C}(T_M^{(i)}) \leq M^\gamma, \text{ for some } \gamma > 0, \quad (1.4)$$

and so the total cost of MC estimator satisfies,

$$\mathcal{C}(\hat{T}_{M,N}^{MC}) \leq NM^\gamma. \quad (1.5)$$

Thus, there are two sources of error in the estimator $\hat{T}_{M,N}^{MC}$:

- the approximation of T by T_M , which is related to the spatial discretisation of our PDE's, referred earlier as bias.
- the sampling error (or statistical error) due to replacing the expected value by a finite sample average.

Let us expand the mean square error (MSE) to see clearly both contributions:

$$\begin{aligned} e(\hat{T}_{M,N}^{MC})^2 &= \mathbb{E} \left[\left(\hat{T}_{M,N}^{MC} - \mathbb{E}[\hat{T}_{M,N}^{MC}] + \mathbb{E}[\hat{T}_{M,N}^{MC}] - \mathbb{E}[T] \right)^2 \right] \\ &= \mathbb{E} \left[(\hat{T}_{M,N}^{MC} - \mathbb{E}[\hat{T}_{M,N}^{MC}])^2 \right] + \left(\mathbb{E}[\hat{T}_{M,N}^{MC}] - \mathbb{E}[T] \right)^2 = \mathbb{V}[\hat{T}_{M,N}^{MC}] + \left(\mathbb{E}[\hat{T}_{M,N}^{MC}] - \mathbb{E}[T] \right)^2, \end{aligned}$$

where \mathbb{E} and \mathbb{V} denote expectation and variance respectively.

Since

$$\mathbb{E}[\hat{T}_{M,N}^{MC}] = \mathbb{E}[T_M] \text{ and } \mathbb{V}[\hat{T}_{M,N}^{MC}] = N^{-1}\mathbb{V}[T_M],$$

we get

$$e(\hat{T}_{M,N}^{MC})^2 = \frac{\mathbb{V}[T_M]}{N} + (\mathbb{E}[T_M] - \mathbb{E}[T])^2, \quad (1.6)$$

and so the first term in the MSE is the variance of the MC estimator, which represents the sampling error and decays inversely with the number of samples. The second term is the square of the error in mean between T_M and T , i.e., the discretisation error.

Hence, a sufficient condition to achieve a RMSE of ε with this estimator is that both of the terms are less than $\varepsilon^2/2$. Under the assumption that $\mathbb{V}[T_M]$ is a constant independent of M , this can be achieved by choosing $N \geq \varepsilon^{-2}$ and $M \geq \varepsilon^{-1/\alpha}$, where the convergence rate, α , defined in (1.1) is problem dependent.

In other words, we need to take a large enough number of samples N , as well as a large enough value for M , so that $\hat{T}_{M,N}^{MC}$ is a sufficiently accurate approximation of our quantity of interest $\mathbb{E}[T]$.

1.2.3 Multilevel Monte Carlo simulation

The main idea of MLMC simulation is to sample not just from one approximation T_M of T , but from several.

Let $\{M_\ell : \ell = 0, \dots, L\}$ be an increasing sequence in \mathbb{N} called *levels*, i.e. $M_0 < M_1 < \dots < M_L =: M$.

As for multi-grid methods applied to discretised (deterministic) PDEs, the key is to avoid estimating $\mathbb{E}[T_{M_\ell}]$ directly on level ℓ , but instead to estimate the correction with respect to the next lower level, i.e., $\mathbb{E}[Y_\ell]$ where $Y_\ell := T_{M_\ell} - T_{M_{\ell-1}}$.

Setting for simplicity $Y_0 := T_{M_0}$ and using the linearity of the expectation operator we have,

$$\mathbb{E}[T_M] = \mathbb{E}[T_{M_0}] + \sum_{\ell=1}^L \mathbb{E}[T_{M_\ell} - T_{M_{\ell-1}}] = \sum_{\ell=0}^L \mathbb{E}[Y_\ell]. \quad (1.7)$$

Hence, the expectation on the finest level is equal to the expectation on the coarsest level, plus a sum of corrections adding the difference in expectation between simulations on consecutive levels. The multilevel idea is now to independently estimate each of these expectations, so that the overall variance is minimised for any given fixed level of computational cost.

Let \hat{Y}_ℓ be an unbiased estimator for $\mathbb{E}[Y_\ell]$, e.g., the standard MC estimator with N_ℓ samples

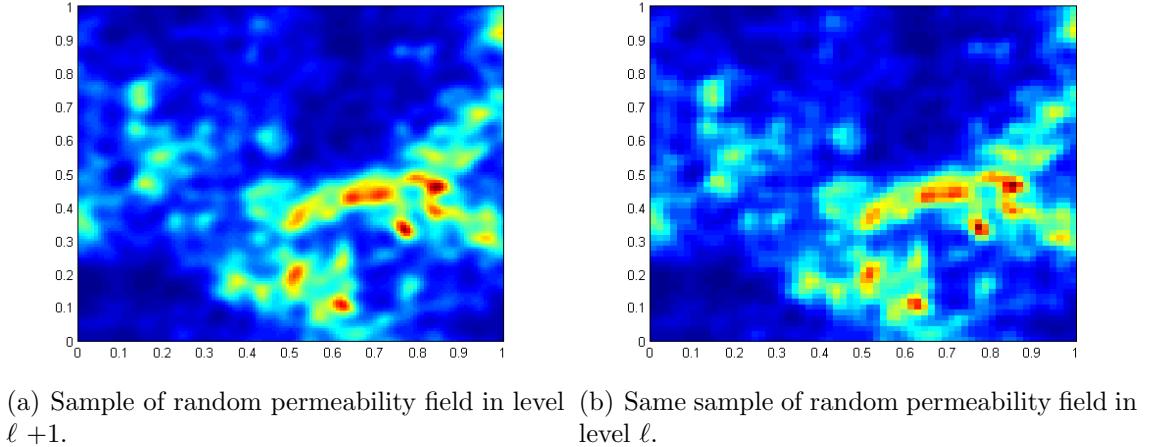
$$\hat{Y}_{\ell, N_\ell}^{MC} := \frac{1}{N_\ell} \sum_{i=1}^{N_\ell} \left(T_{M_\ell}^{(i)} - T_{M_{\ell-1}}^{(i)} \right), \quad (1.8)$$

then the multilevel estimator is defined as

$$\hat{T}_M^{ML} := \sum_{\ell=0}^L \hat{Y}_\ell. \quad (1.9)$$

If the individual terms are estimated using standard MC, $\hat{Y}_{\ell, N_\ell}^{MC}$, with N_ℓ samples on level ℓ , this is the multilevel Monte Carlo estimator and we denote it by $\hat{T}_{M, \{N_\ell\}}^{MLMC}$.

Note that the quantity $T_{M_\ell}^{(i)} - T_{M_{\ell-1}}^{(i)}$ must be calculated from the same random sample $\omega^{(i)} \in \Omega$, i.e., we use a coarsened version of the same input used for $T_{M_\ell}^{(i)}$ in calculating $T_{M_{\ell-1}}^{(i)}$. The physical meaning of this is showed in Figure 1.2.



(a) Sample of random permeability field in level $\ell + 1$. (b) Same sample of random permeability field in level ℓ .

Figure 1.2: Two samples of the same random permeability field in two consecutive levels to be used as input in the MLMC method.

Since all the expectations $\mathbb{E}[\hat{Y}_\ell]$ are estimated independently, the variance of the MLMC estimator is $\mathbb{V}[\hat{T}_M^{ML}] = \sum_{\ell=0}^L N_\ell^{-1} \mathbb{V}[Y_\ell]$, and so the MSE is

$$e(\hat{T}_M^{ML})^2 := \mathbb{E} [(\hat{T}_M^{ML} - \mathbb{E}[T])^2] = \sum_{\ell=0}^L \frac{\mathbb{V}[Y_\ell]}{N_\ell} + (\mathbb{E}[T_M - T])^2. \quad (1.10)$$

As in the standard MC case before, we see that the MSE consists of two terms, the variance of the estimator and the approximation error. To bound the RMSE by ε , we can again seek to bound each term above by $\varepsilon/2$. Note that the second term is exactly the same as in Equation (1.6) and so it is sufficient to choose $M = M_L \geq \varepsilon^{-1/\alpha}$ again. And therefore, to then achieve an overall RMSE of ε ,

the first term of $e(\hat{T}_M^{ML})^2$ has to be less than $\varepsilon^2/2$ as well.

The computational cost of the MLMC estimator is

$$\mathcal{C}(\hat{T}_M^{ML}) = \sum_{\ell=0}^L N_\ell \mathcal{C}_\ell. \quad (1.11)$$

where $\mathcal{C}_\ell := \mathcal{C}(Y_\ell^{(i)})$ represents the cost of a single sample of Y_ℓ .

The variance of MLMC estimator can be minimised (Cliffe et al., 2011) for a fixed computational cost by choosing

$$N_\ell \simeq \sqrt{\mathbb{V}[Y_\ell] \mathcal{C}_\ell}, \quad (1.12)$$

with the constant of proportionality chosen so that the overall variance is $\varepsilon^2/2$. So, the total cost on level ℓ is proportional to $\sqrt{\mathbb{V}[Y_\ell] \mathcal{C}_\ell}$ and hence

$$\mathcal{C}(\hat{T}_M^{ML}) \leq \sum_{\ell=0}^L \sqrt{\mathbb{V}[Y_\ell] \mathcal{C}_\ell}. \quad (1.13)$$

- If the variance $\mathbb{V}[Y_\ell]$ decays faster with ℓ than \mathcal{C}_ℓ increases, the dominant term will be on level 0. Since $N_0 \simeq \varepsilon^{-2}$, the cost savings compared to standard MC will in this case be approximately $\frac{\mathcal{C}_0}{\mathcal{C}_L} \simeq \left(\frac{M_0}{M_L}\right)^\gamma \simeq \varepsilon^{\gamma/\alpha}$, reflecting the ratio of the costs of samples on level 0 compared to samples on level L .
- If the variance $\mathbb{V}[Y_\ell]$ decays slower than the cost \mathcal{C}_ℓ increases, the dominant term will be on the finest level L , and the cost savings compared to standard MC will be approximately $\mathbb{V}[Y_L]/\mathbb{V}[Y_0]$ which is $\mathcal{O}(\varepsilon^2)$, if we have truncated the telescoping sum $\mathbb{E}[T_M] = \mathbb{E}[T_{M_0}] + \sum_{\ell=1}^L \mathbb{E}[T_{M_\ell} - T_{M_{\ell-1}}] = \sum_{\ell=0}^L \mathbb{E}[Y_\ell]$ with M_0 such that $\mathbb{V}[Y_0] \simeq \mathbb{V}[T_0]$.

Hence, in both cases we have a significant gain.

The optimal values of L and $\{N_\ell\}_{\ell=0}^L$ referred earlier can be computed from the sample averages and the unbiased sample variances of Y_ℓ .

If we assume that $|\mathbb{E}[T_M - T]| \simeq M^{-\alpha}$, then it follows that $|\mathbb{E}[Y_\ell]| \simeq M^{-\alpha}$ and $|\mathbb{E}[\hat{Y}_L]| \simeq M^{-\alpha}$ for N_L sufficiently large providing us with a computable error

estimator to determine whether M is sufficiently large or whether the number of levels L needs to be increased.

The above assumptions and statements are formally presented in the following Theorem (Cliffe et al., 2011):

Theorem 1.2.1 *Let $\hat{Y}_\ell := \hat{Y}_{\ell, N_\ell}^{MC}$ and suppose that there are positive constants $\alpha, \beta, \gamma, C_\alpha, C_\beta, C_\gamma > 0$ such that $\alpha \geq \frac{1}{2} \min(\beta, \gamma)$ and*

$$1. |\mathbb{E}[T_{M_\ell} - T]| \leq C_\alpha M_\ell^{-\alpha}$$

$$2. \mathbb{V}[Y_\ell] \leq C_\beta M_\ell^{-\beta}$$

$$3. \mathcal{C}_\ell \leq C_\gamma M_\ell^{-\gamma},$$

Then, for any $\varepsilon < e^{-1}$, there exist a positive constant C^{ML} , a value L (and corresponding $M \equiv M_L$) and a sequence $\{N_\ell\}_{\ell=0}^L$ such that

$$e(\hat{T}_M^{ML})^2 := \mathbb{E} \left[(\hat{T}_M^{ML} - \mathbb{E}[T])^2 \right] < \varepsilon^2,$$

and

$$\mathcal{C}(\hat{T}_M^{ML}) = \begin{cases} C^{ML} \varepsilon^{-2}, & \text{if } \beta > \gamma, \\ C^{ML} \varepsilon^{-2} (\log \varepsilon)^2, & \text{if } \beta = \gamma, \\ C^{ML} \varepsilon^{-2-(\gamma-\beta)/\alpha}, & \text{if } \beta < \gamma. \end{cases}$$

Whereas

$$\mathcal{C}(\hat{T}_M^{MC}) = C^{MC} e^{-2-\gamma/\alpha},$$

for some positive constant C^{MC} .

Proof The proof is given in (Cliffe et al., 2011), Appendix A.

The MLMC algorithm can be implemented in practice as follows:

1. Start at the coarsest level ($L=0$).
2. Estimate $\mathbb{V}[Y_L]$ by the sample variance of an initial number of N_L samples.

Remember that $Y_0 := T_{M_0}$, i.e, quantity of interest in level 0 (coarsest level) and $Y_\ell := T_{M_\ell} - T_{M_{\ell-1}}$.

3. Calculate the optimal N_ℓ , $\ell = 0, \dots, L$, using (1.12), Remember that $\mathcal{C}_\ell := \mathcal{C}(Y_\ell^{(i)})$ represents the cost of a single sample of Y_ℓ .

This step aims to make the variance of the MLMC estimator (1.9) less than $\frac{1}{2}\varepsilon^2$.

4. Evaluate extra samples at each level as needed for the new N_ℓ .

5. If $L \geq 1$, test for convergence using $\hat{Y}_L \simeq M^{-\alpha}$.

Remember that $\hat{Y}_\ell = \hat{Y}_{\ell, N_\ell}^{MC} := \frac{1}{N_\ell} \sum_{i=1}^{N_\ell} (T_{M_\ell}^{(i)} - T_{M_{\ell-1}}^{(i)})$

This step tries to ensure that the remaining bias ($\mathbb{E}[T_M - T]$) is less than $\frac{1}{\sqrt{2}}\varepsilon$.

6. If not converged, set $L = L + 1$ and go back to 2.

The parameters, α , β and γ that can be estimated empirically as follows:

For γ , we assume that the number of operations to compute one sample on level ℓ is $\mathcal{C}_\ell = cM_\ell^\gamma$ for some constant c independent of ℓ . For β , we can use as an approximation the slope of the line for $\mathbb{V}[Y_\ell]$, m_β , because $\mathbb{V}[Y_\ell] \simeq M_\ell^{-m_\beta}$. For α , we can use as an approximation the slope of the line for $\mathbb{E}[T_\ell - T_{\ell-1}]$, m_α , because $\mathbb{E}[T_\ell - T_{\ell-1}] \simeq M_\ell^{-m_\alpha}$.

1.2.4 Quasi Monte Carlo simulation

The MC method used to solve the system of PDEs with random coefficients is based on pseudo-random number sampling algorithms. So in this case, during the process, uniformly distributed pseudo-random numbers are generated and

transformed into M -dimensional points that are distributed according to a certain probability distribution. These points are used to form the inputs (e.g., random coefficients representing the permeability fields) required by the simulator in order to compute the corresponding output (e.g. pressure field, travel time etc.). Sometimes, two input values are very close together, and, in this case, it might not be necessary to run the simulator at both inputs since the output will be practically the same (note that we are dealing with a deterministic simulator). Let us see an illustrative example: Suppose we wish to compute $I = \int_{[0,1]^M} f(\mathbf{x})d\mathbf{x}$ with the MC method. Let $f : [0, 1]^M \rightarrow \mathbb{R}$, and $Y = f(X)$, where X is uniformly distributed in $[0, 1]^M$. Let p denote the uniform probability density function and letting x be uniformly distributed in $[0, 1]^M$, we can apply MC quadrature to approximate I , for a given $N \in \mathbb{N}$, in the following way,

$$I = \int_{[0,1]^M} f(\mathbf{x})d\mathbf{x} = \int_{[0,1]^M} f(\mathbf{x})p(\mathbf{x})d\mathbf{x} = \mathbb{E}[f(\mathbf{x})] \simeq \frac{1}{N} \sum_{j=1}^N f(\mathbf{x}(\omega_j)) = I_N$$

where the values $\mathbf{x}(\omega_j)$ are independent and identically distributed (iid) random variables sampled uniformly in the cube $[0, 1]^d$ by sampling the components $x_i(\omega_j)$ independently and uniformly on the interval $[0, 1]$.

QMC aims to choose adequate locations of the previous pseudo-random numbers (ω_j in the previous example) in a deterministic manner, in such a way, to guarantee that the points are uniformly spaced in the random space in order to obtain the maximum possible information about the quantity of interest with the minimum number of simulator runs, and therefore optimizing the MC method. The generation of these pseudo-random numbers can be done by using, for instance, digital nets (J.Dick and F.Pillichshammer, 2010), rank-1 lattice rule (J.Hickernell and H.Niederreiter, 2003) or Sobol sequences (Sobol, 1967). We will use Sobol sequences in this work to be consistent with the method used to build the design of our Gaussian process emulator.

In practical terms, to apply the QMC method we first generate a Sobol sequence which is space filling on $[0, 1]^M$, where M is the required dimension of each of the points. Then, those points are pushed through the inverse cumulative distribution function of a random variable. These new random vectors are used

as the inputs for our simulator.

The biggest difference to pseudo-random numbers is that the sample values are chosen under consideration of the previously sampled points, thus avoiding the occurrence of clusters and gaps, as we can observe in the plots below. Figure 1.3(g) shows 100 pseudo-random numbers generated from a uniform distribution and Figure 1.3(h) same number of points generated by Sobol sequences; we can observe some more gaps (white spaces) in 1.3(g) than in 1.3(h) where it seems that the sampling space is filled in a more uniformly manner. Figure 1.3(a) and 1.3(b) show the distribution of 2000 points with the two approaches; we can appreciate bigger gaps and some clusters (dark blue stains) of points in Figure 1.3(a) than in 1.3(b).

The QMC estimator used for estimating $\mathbb{E}(T_M)$ in this case is defined as,

$$\hat{T}_{M,N}^{QMC} := \frac{1}{N} \sum_{i=1}^N T_{Q,M}^{(i)}, \quad (1.14)$$

where $T_{Q,M}^{(i)}$ is the i th sample of T_M generated from QMC inputs, and N independent samples are computed in total.

1.2.5 Multilevel Quasi Monte Carlo simulation

This method is a straightforward consequence of applying the MLMC algorithm with a QMC estimator for each of the levels instead of MC estimator.

Thus, if we now let \hat{Y}_ℓ be the unbiased QMC estimator for $\mathbb{E}[Y_\ell]$ with N_ℓ samples

$$\hat{Y}_{\ell,N_\ell}^{QMC} := \frac{1}{N_\ell} \sum_{i=1}^{N_\ell} \left(T_{M_\ell}^{(i)} - T_{M_{\ell-1}}^{(i)} \right), \quad (1.15)$$

then use the multilevel estimator defined in (1.9),

$$\hat{T}_M^{ML} := \sum_{\ell=0}^L \hat{Y}_\ell. \quad (1.16)$$

If the individual terms are estimated using QMC, $\hat{Y}_{\ell,N_\ell}^{QMC}$, with N_ℓ samples on

level ℓ , this is the multilevel Quasi Monte Carlo estimator and we denote it by $\hat{T}_{M,\{N_\ell\}}^{MLQMC}$.

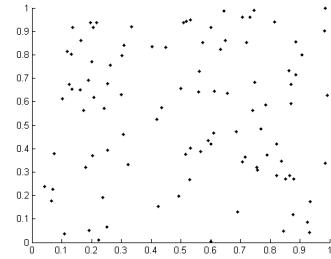
The MLQMC algorithm can be implemented in practice by following the next steps:

1. Start $L=0$.
2. Estimate $\mathbb{V}[Y_L]$ by the sample variance of an initial number of N_L samples.
3. Calculate the optimal N_ℓ , $\ell = 0, \dots, L$, using (1.12)
4. Evaluate extra samples at each level as needed for the new N_ℓ .

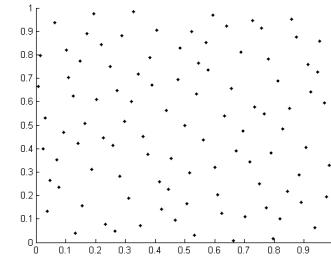
Note that in this step, special care has to be taken when evaluating extra samples since QMC estimator is built in a deterministic manner and the purpose here is to add new information to reduce the sample variance.

5. If $L \geq 1$, test for convergence using $\hat{Y}_L \simeq M^{-\alpha}$.
6. If not converged, set $L = L + 1$ and go back to 2.

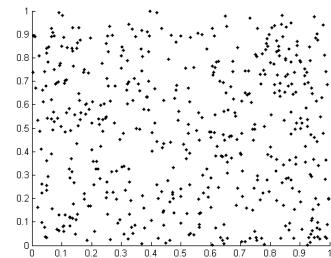
Complex simulators based on mechanistic and physical processes are often computationally expensive, and full UA becomes extremely time consuming, if not unfeasible. One solution to this is to create an statistic surrogate model for the simulator. This surrogate is typically modelled as a GP and its aim is to obtain an accurate approximation of the simulator by just using a finite number of simulator runs. The proposed emulation methodology will focus on emulating two statistics of the the model: the mean and the distribution function of the quantity of interest. The mean is a straightforward statistic to emulate and provides some information about how the output changes due to changes in each input. The distribution function is more complex to estimate, however, it is an important statistic since it contains information about the entire distribution of the outputs.



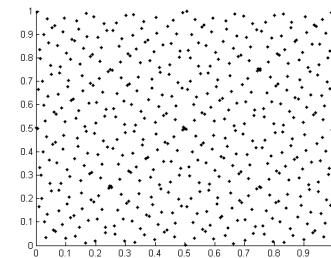
(a) 100 two-dimensional pseudo-random numbers generated uniformly over the unit square.



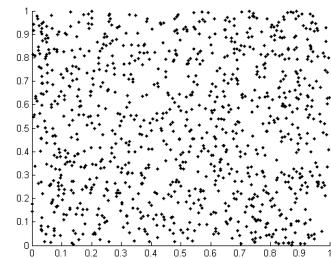
(b) 100 two-dimensional numbers generated by Sobol sequences over the unit square.



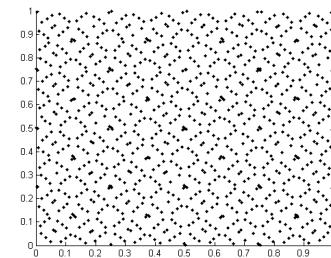
(c) 500 two-dimensional pseudo-random numbers generated uniformly over the unit square.



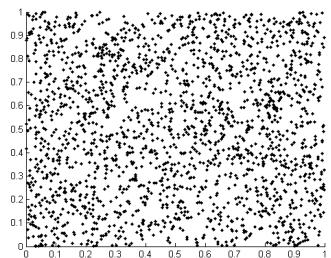
(d) 500 two-dimensional numbers generated by Sobol sequences over the unit square.



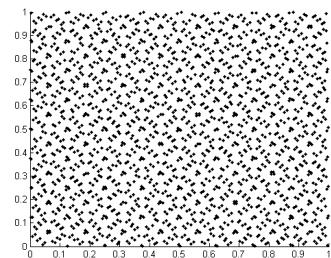
(e) 1000 two-dimensional pseudo-random numbers generated uniformly over the unit square.



(f) 1000 two-dimensional numbers generated by Sobol sequences over the unit square.



(g) 2000 two-dimensional pseudo-random numbers generated uniformly over the unit square.



(h) 2000 two-dimensional numbers generated by Sobol sequences over the unit square.

Figure 1.3: Pseudo-random and Sobol sequences based sampling comparison over the unit square.

1.3 GP emulation of computer models

We now introduce an alternative approach to MC simulation methods. This consists of building a statistical approximation of the simulator, which we will call *the emulator*. We use Bayesian methodology to build the emulator, specifying a prior belief about the functional form of the model, that is updated in the light of the training data, in order to give the *posterior distribution* for the function. In general, a posterior distribution can be obtained from the *prior distribution* and the *likelihood* by applying Bayes rule in the form,

$$\text{Posterior} = \frac{\text{Likelihood} \times \text{Prior}}{\text{Normalizing constant}}.$$

We can then say that the *posterior* is the *prior* conditioned on the observations provided. The observations are called the *training set*, and will be denoted by $\mathcal{D} = \{(\mathbf{x}_i, y_i); i = 1, \dots, d\}$, where each \mathbf{x}_i denotes an input vector of dimension D and $y_i = f(\mathbf{x}_i)$ denotes a scalar output (sometimes called *the target*).

Because we have previously considered the simulator as a function $f(\cdot)$ that maps inputs \mathbf{x} into an output $y = f(\mathbf{x})$, we could imagine using an approximation $\hat{f}(\cdot)$ instead of $f(\cdot)$, e.g., for the UA calculations. If the approximation is good enough, then the uncertainty measure produced by the analysis will be sufficiently close to those that would have been obtained using the original simulator $f(\cdot)$.

In general, an emulator, also called a meta-model, only requires a small number of runs of the expensive simulator in order to achieve a desired performance, making the process computationally cheaper than carrying out the full Monte Carlo analysis. These runs are called the *training runs* of the model, in which $y_1 = f(\mathbf{x}_1), y_2 = f(\mathbf{x}_2), \dots, y_d = f(\mathbf{x}_d)$ are *observed* and used to estimate $f(\mathbf{x})$. Due to the computational cost, we wish to carefully choose the points, $\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_d\}$, called the *design points*, and for them compute the corresponding values $\{y_1 = f(\mathbf{x}_1), y_2 = f(\mathbf{x}_2), \dots, y_d = f(\mathbf{x}_d)\}$ to form the training data used to update the prior. We will discuss the choice of design points further in Section 2.1.

In order to build a successful emulator it is useful to consider what O'Hagan (2004) presents as two natural criteria that the emulator should satisfy: First, at

a design point \mathbf{x}_i , the emulator should reflect the fact that we know the true value of the simulator output, so it should return $\hat{f}(\mathbf{x}_i) = y_i$ with no uncertainty. And second, at other points, the distribution for $f(\mathbf{x})$ should give a mean value $\hat{f}(\mathbf{x})$ that represents a plausible interpolation or extrapolation of the training data, and the probability distribution around this mean should be a realistic expression of uncertainty about how the simulator might interpolate/extrapolate.

To finish with this introduction, where we have described all theoretical aspects of the mathematical tools which we will be used later in this thesis, in the following section we will give an overview of GP regression models.

1.3.1 GP regression

A GP is a collection of random variables, any finite number of which have a joint Gaussian distribution. There are two equivalent ways to interpret GP regression models, the weight-space view and the function-space view (Rasmussen and Williams, 2006). In this thesis we will follow the second approach, i.e., we will think of a Gaussian process as defining a distribution over functions, and inference taking place directly in the space of functions.

A Gaussian process is completely specified by its mean function and covariance function. This is a natural generalization of the Gaussian distribution whose mean and covariance is a vector and matrix, respectively. The Gaussian distribution is over vectors, whereas the Gaussian process is over functions. We define the mean function, $m(\mathbf{x})$, and the covariance function, $k(\mathbf{x}, \mathbf{x}')$, of a real process $f(\mathbf{x})$ in terms of its expectation as:

$$m(\mathbf{x}) = \mathbb{E}[f(\mathbf{x})],$$

and

$$k(\mathbf{x}, \mathbf{x}') = \mathbb{E}[(f(\mathbf{x}) - m(\mathbf{x}))(f(\mathbf{x}') - m(\mathbf{x}'))],$$

and denote the Gaussian process as:

$$f(\mathbf{x}) \sim \mathcal{GP}(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')). \quad (1.17)$$

Given any set of s points, $\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_s\}$, the vector $\{f(\mathbf{x}_1), f(\mathbf{x}_2), \dots, f(\mathbf{x}_s)\}$ has a multivariate Gaussian distribution with mean vector, $\boldsymbol{\mu}$, and covariance matrix, Σ , where,

$$\mu_i = m(\mathbf{x}_i), \quad i = 1, \dots, s,$$

and

$$\Sigma_{ij} = k(\mathbf{x}_i, \mathbf{x}_j), \quad i, j = 1, \dots, s,$$

If we denote the vector, \mathbf{f} , where $f_i = f(x_i)$, then

$$\mathbf{f} \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma).$$

We will show next how we can use GP regression to approximate our simulator.

1.3.2 Building the GP emulator

As discussed in Section 1.3, given a GP, it can be used as a prior specification for Bayesian inference. The specification of the prior fixes the properties of the functions considered for inference and, in this case, our prior specification for the emulator is that we model f as a Gaussian process with mean function m , and covariance function k , as stated in expression (1.17).

A key property of GPs that makes them so successful as modelling tools, is that their posterior distribution, after training them on the training ensemble \mathcal{D} , is still a GP, in this case, given a prior as in expression (1.17) and a training set \mathcal{D} , we have:

$$f(\mathbf{x})|_{\mathcal{D}} \sim \mathcal{GP}(m_{\mathcal{D}}(\mathbf{x}), k_{\mathcal{D}}(\mathbf{x}, \mathbf{x}')), \quad (1.18)$$

with updated mean and covariance functions, $m_{\mathcal{D}}$ and $k_{\mathcal{D}}$.

So, if we pass to the GP our beliefs of a suitable prior distribution by providing it with the mean, covariance, likelihood functions and adequate training set to update that prior, we can then use the GP emulator to make predictions. In the regression setting the outputs are real values. More formally, we are interested in making inferences about the relationship between inputs and outputs, i.e., the conditional distribution of the outputs given the inputs as we will see in detail next in Section 1.3.3.

1.3.3 Prediction using the GP emulator

Given a set of points $\mathbf{x} = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}$ we can run our simulator, f , at these locations in order to obtain the corresponding outputs $y_i = f(\mathbf{x}_i)$. It is typical for more realistic modelling situations that we do not have access to function values themselves, but only noisy versions in the form¹,

$$\mathbf{y} = f(\mathbf{x}) + \boldsymbol{\epsilon},$$

where $\boldsymbol{\epsilon}$ is an i.i.d Gaussian noise with variance σ_n^2 , i.e., $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \sigma_n^2 \mathbf{I})$. This noise can be due, for instance, to some truncation error made by the simulator or just because of the rounding error of some parameter of the model. So, with these considerations about the outputs, and following the notation for the mean and covariance established in Section 1.3.1, the prior on the noisy observations now becomes,

$$\mu_i = m(\mathbf{x}_i),$$

and

$$\text{cov}(y_i, y_j) = k(\mathbf{x}_i, \mathbf{x}_j) + \sigma_n^2 \delta_{ij},$$

where cov is the covariance and δ_{ij} is the Kronecker delta ($\delta_{ij} = 1$ iff $i = j$ and 0

¹Given a set of points $\mathbf{x} = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}$ and a realisation of $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \sigma_n^2 \mathbf{I})$, $\boldsymbol{\epsilon} = \{\epsilon_1, \dots, \epsilon_n\}$, we can take $\mathbf{y} = \{y_1, \dots, y_n\}$ as the noisy observations, where $y_i = f(\mathbf{x}_i) + \epsilon_i$ for $i = 1, \dots, n$.

otherwise). In matrix form we have,

$$cov(\mathbf{y}) = \Sigma(\mathbf{X}, \mathbf{X}) + \sigma_n^2 \mathbf{I},$$

where $\Sigma(\mathbf{X}, \mathbf{X})$ denotes the $(d \times d)$ matrix of the covariances evaluated at all pairs of training, \mathbf{x} . If we consider a training set $\mathcal{D} = \{(\mathbf{x}_i, y_i); \mathbf{x}_i \in \mathbb{R}^D, y_i \in \mathbb{R}, i = 1, \dots, d\}$, we can form now the $D \times d$ *design matrix*, \mathbf{X} , and write the training set as $\mathcal{D} = \{(\mathbf{X}, \mathbf{y})\}$. We can write now the joint distribution of the noisy output values, \mathbf{y} , and the predicted values, \mathbf{f}_* , at the test locations, \mathbf{x}_* , under the prior as,

$$\begin{bmatrix} \mathbf{y} \\ \mathbf{f}_* \end{bmatrix} \sim \mathcal{N} \left(\begin{bmatrix} \boldsymbol{\mu} \\ \boldsymbol{\mu}_* \end{bmatrix}, \begin{bmatrix} \Sigma(\mathbf{X}, \mathbf{X}) + \sigma_n^2 \mathbf{I} & \Sigma(\mathbf{X}, \mathbf{X}_*) \\ \Sigma(\mathbf{X}_*, \mathbf{X}) & \Sigma(\mathbf{X}_*, \mathbf{X}_*) \end{bmatrix} \right), \quad (1.19)$$

where $\boldsymbol{\mu}_*$ is a vector formed by the test means, and, since there are d training points, if we let d_* be the number of test points $\{\mathbf{x}_{*_1}, \mathbf{x}_{*_2}, \dots, \mathbf{x}_{*_d}\}$, then $\Sigma(\mathbf{X}, \mathbf{X}_*)$ denotes the $(d \times d_*)$ matrix of the covariances evaluated at all pairs of training, \mathbf{x} , and test points, \mathbf{x}_* , and similarly for the other entries $\Sigma(\mathbf{X}, \mathbf{X})$, $\Sigma(\mathbf{X}_*, \mathbf{X}_*)$ and $\Sigma(\mathbf{X}_*, \mathbf{X}_*)$. Deriving the corresponding conditional distribution¹ we arrive at the key predictive equations for Gaussian process regression (Rasmussen and Williams, 2006),

$$\mathbf{f}_* | \mathbf{X}, \mathbf{y}, \mathbf{X}_* \sim \mathcal{N}(\bar{\mathbf{f}}_*, cov(\mathbf{f}_*)),$$

where

$$\bar{\mathbf{f}}_* := \mathbb{E}[\mathbf{f}_* | \mathbf{X}, \mathbf{y}, \mathbf{X}_*] = \boldsymbol{\mu} + \Sigma(\mathbf{X}_*, \mathbf{X}) [\Sigma(\mathbf{X}, \mathbf{X}) + \sigma_n^2 \mathbf{I}]^{-1} \mathbf{y}, \quad (1.20)$$

and

$$cov(\mathbf{f}_*) = \Sigma(\mathbf{X}_*, \mathbf{X}_*) - \Sigma(\mathbf{X}_*, \mathbf{X}) [\Sigma(\mathbf{X}, \mathbf{X}) + \sigma_n^2 \mathbf{I}]^{-1} \Sigma(\mathbf{X}, \mathbf{X}_*). \quad (1.21)$$

If we now consider a single test input, \mathbf{x}_{*i} , and we let \mathbf{k}_* be the vector of covariances between the test point and the d training points, $\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_d\}$, we can reduce the equations for predictions (1.20) and (1.21) to a single test case in the

¹See Rasmussen and Williams (2006), Appendix A for further details of Gaussian identities.

form,

$$\bar{f}_{*_i} = \mu_{*_i} + \mathbf{k}_*^\top [\Sigma(\mathbf{X}, \mathbf{X}) + \sigma_n^2 I]^{-1} \mathbf{y}, \quad (1.22)$$

and

$$\mathbb{V}ar(f_{*_i}) = k_{*_i} - \mathbf{k}_*^\top [\Sigma(\mathbf{X}, \mathbf{X}) + \sigma_n^2 I]^{-1} \mathbf{k}_*, \quad (1.23)$$

where $\mu_{*_i} = m(\mathbf{x}_{*_i})$, $k_{*_i} = k(\mathbf{x}_{*_i}, \mathbf{x}_{*_i})$, and $\mathbb{V}ar$ is the variance.

Let us finish this introductory chapter with an illustrative example where we show how we use GP regression to approximate a simple one dimensional deterministic function.

1.3.4 An Illustrative one dimensional example

Let us consider the deterministic function $f(x) = 1 + \frac{3}{2} \sin x + x^2$, for $x \in [-4, 4]$. The goal is to use GP regression to approximate f .

First, we consider the GP given by:

$$f(x) \sim \mathcal{GP}(m(x), k(x, x')), \text{ where } m(x) = \frac{1}{4}x^2, \text{ and } k(x, x') = \exp\left(-\frac{1}{2}(x - x')^2\right).$$

This means, in other words, that we want to model f as a GP with these explicit mean and covariance function m and k . Samples¹² from the prior distribution of the function f are shown in Figure 1.4(a). As we want to work only with finite quantities, we request only the value of f at a distinct finite number of locations (therefore dotted lines) within the interval $[-4, 4]$. Now we provide a training set of 6 pairs of points, Figure 1.4(b), and update the prior in the light of this training set, Figure 1.4(c). The whole picture with the difference between prior and posterior samples is presented in Figure 1.4(d). Note in this later case the

¹For simulating samples, f_i , from a Gaussian process we use the following formula $f_i(\mathbf{x}) = m(\mathbf{x}) + \text{chol}(k(\mathbf{x}, \mathbf{x})) \mathbf{u}$, where chol represents the Cholesky factorization of the matrix associated to the covariance function k evaluated at a set of points $\mathbf{x} \in [-4, 4]$, and \mathbf{u} is an i.i.d. random vector $\mathcal{N}(\mathbf{0}, \mathbf{I})$.

²The dots in Figure 1.4(a) are the values generated from the prior distribution, the other curves representing the posterior samples in Figure 1.4(c) have (less correctly) been drawn by connecting sampled points.

difference between our prior assumption (prior samples) of the functional form of f and the final model (posterior samples) obtained in the light of the 6 observed points supplied. We can derive and use the posterior mean and variance to represent the uncertainty we have with respect to the true f , Figure 1.5(a), in this case corresponding to 95% confidence intervals. Now we can increase the size of the training set and provide more observations and see how the uncertainty region is reduced, Figure 1.5(b).

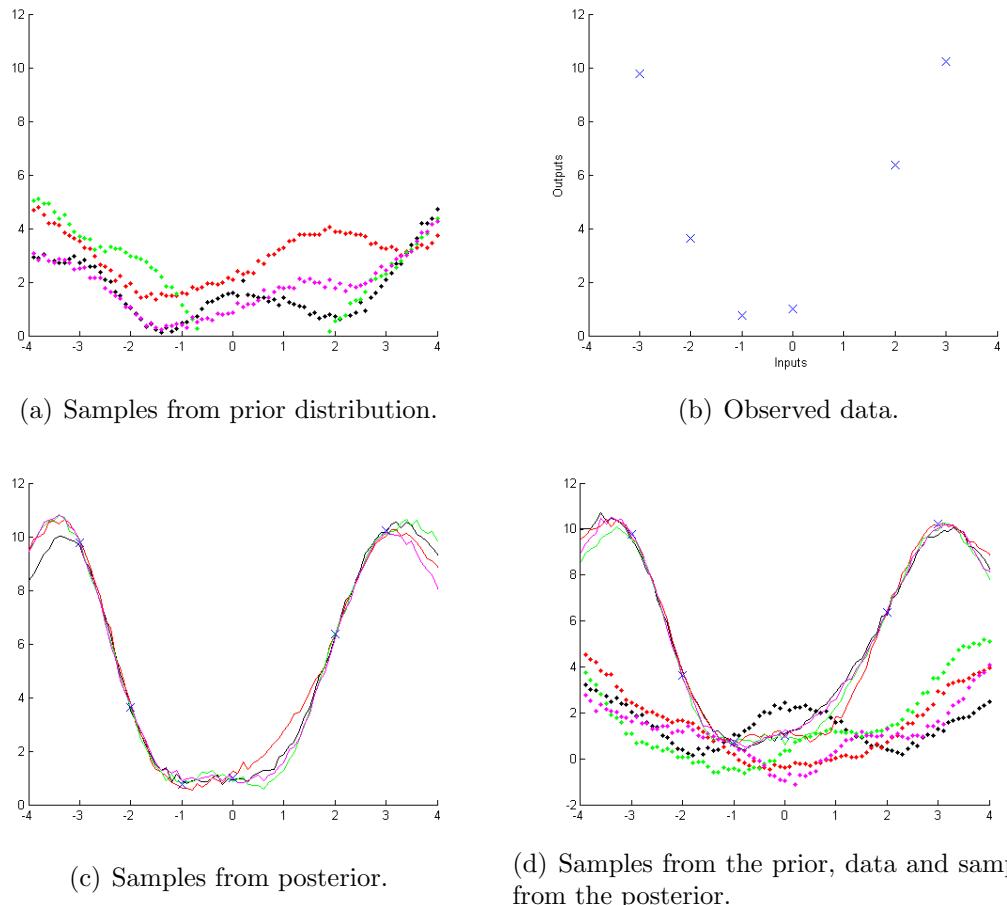
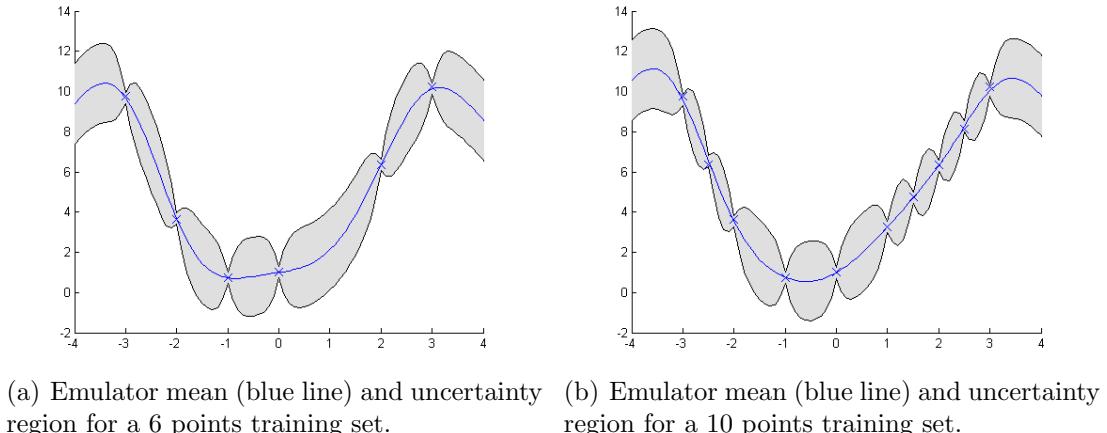


Figure 1.4: Gaussian process regression example. (a) Samples drawn from the GP prior distribution with mean $m(x) = \frac{1}{4}x^2$ and covariance $k(x, x') = \exp\left(-\frac{1}{2}(x - x')^2\right)$. (b) A training set of 6 points. (c) Samples drawn from the posterior distribution. (d) Differences between prior samples and the updated posterior in the light of the training set.



(a) Emulator mean (blue line) and uncertainty region for a 6 points training set.

(b) Emulator mean (blue line) and uncertainty region for a 10 points training set.

Figure 1.5: Emulator approximation to $f(x) = 1 + \frac{3}{2} \sin x + x^2$, and uncertainty region corresponding to 95% confidence intervals for different training sets. We notice how the uncertainty region is negligible around training points and it is reduced when including more points to the training set.

1.4 Outline of the thesis

The aim of this thesis is to study and analyse current computational tools for performing uncertainty quantification in groundwater flow and convectively-enhanced dissolution (C-ED) processes. Within this scope, this work focuses on accelerating the modelling and simulation process by substituting simpler, dynamically generated, computationally cheap surrogate models in place of the full evaluation of the computational model. Our goal is to build a competitive surrogate model, the GP emulator, that work well not only for simple simulators but for more complex industry problems. The election of the first model, the single Darcy's flow model, is due to the fact that this model has been studied deeply by the scientific community (see e.g., Cliffe et al. (2011)) and therefore this provides us with a benchmark in order to test the efficiency and accuracy of our GP emulator. So far, we have broadly introduced all the methodologies that are being used by the scientific community for doing uncertainty quantification

in computer models. In the following chapters of this thesis we will put all these methodologies into practice.

In Chapter 2 we introduce the deterministic simulator used for the single Darcy's flow model. Then we convert the deterministic simulator in a system of PDEs with random coefficients by modelling one of the simulator input parameters, the permeability, as a lognormal random field. We describe how we build our GP emulator for the new simulator and show how we use cross-validation methods to test the emulator performance. We use the GP emulator to approximate the cumulative distribution function (CDF) of the simulator scalar output and compare the results with the ones obtained by applying the standard Monte Carlo simulation method. We conclude the chapter by describing and applying a novel methodology, based in the use of GP emulation and singular value decomposition (SVD) algorithms, for approximating field outputs instead of scalar outputs.

In Chapter 3 we use the GP emulator, MC, MLMC, QMC and MLQMC methods to approximate and compare the results for the average travel time of a particle travelling from the centre of a given physical domain to get to the domain boundary. A series of experiments based on several numerical discretisations of the physical domain are conducted. For each of these discretisations we apply all the methodologies proposed in the introduction of this thesis, namely, GP emulator, MC, MLMC, QMC and MLQMC and compare the outcomes. The results obtained with the GP emulator are given in terms of confidence intervals (CI). The performance of the MC methods is tested by comparing their computational ε -cost, i.e. the number of floating point operations that are needed to achieve the desired MSE.

In Chapter 4 we describe the deterministic simulator for the C-ED process. This is a more physical and computational complex real problem than the previous one discussed in Chapter 2, and therefore, the search of solutions for the deterministic model is not straightforward. In order to find numerical solutions for this problem we present a novel method based of the finite element (FE) method and arclength continuation techniques. In this model, the uncertain input of the simulator is again permeability, and so we use the same method to model the permeability as in chapter 2. We describe how we build our GP emulator for the

simulator with random inputs. We use the GP emulator to approximate the CDF of the simulator scalar output and compare the results with the ones obtained by applying the standard Monte Carlo simulation method. We conclude the chapter by applying the same methodology introduced in chapter 2 to approximate field outputs instead of scalar outputs.

In Chapter 5 we discuss conclusions that have arisen from the application of the broad range of methodologies to groundwater flow and C-ED problems, and draw lines of potential areas of future research.

Finally, after Chapter 5, we show a list of all the abbreviations and a list of all the symbols used throughout this thesis.

Chapter 2

Gaussian process emulation for uncertainty quantification in groundwater flow models

We start this chapter by describing how we build the GP emulator for a general simulator and how we test it by following cross-validation methods. We then introduce the simulator used in this thesis to solve a particular groundwater flow model. We detail how we model the uncertain input parameter of this model and show how the emulator can be used to quantify the uncertainty distribution of the simulator scalar output. We finish the chapter with the description of a novel technique, based on the combination of GP emulation and SVD methodologies, for emulating two dimensional outputs instead of scalar outputs, and compare this approach with the standard direct emulation over scalar outputs.

The GP emulator presented in this thesis has been built upon the Matlab¹ code GPML developed by Rasmussen and Williams (2006). As discussed in Section 1.3, the Bayesian methodology is based on a prior belief about the functional form of the model, that is updated in the light of the training data in order to give the posterior distribution for the function to approximate. Thus, another key point for a GP emulator to be successful is an adequate selection of the training data used to update the prior assumptions. Next, we are going to describe, first, how we choose our design, and second, the type of mean and covariance functions that we will use in our GP emulator.

2.1 Generation of the design points

As we need a reduced number of simulator runs providing the maximum possible information about the output, we need to choose a design which covers the full range of uncertainty about the input values. For instance, if two input values are very close together, there may be some issues when building the emulator, as within the process, some of the covariance matrices involved may become singular and therefore cannot be inverted. The design points do not need to be random. The objective is to learn about the function $f(\cdot)$, and well spaced points that cover the region of interest are much better than random points. Sampling is the process of exploring the domain of interest. A random sample can be generated by a pseudo-random number generator. A sample is randomly distributed in a defined interval according to some distribution. There are several methods of sampling the input values as the Latin Hypercube Sampling (LHS) (McKay et al., 1979) described by Pebesma and Heuvelink (1999) or Sobol sequence based method (Sobol, 1967). We will use the second to build our design by generating the pseudo-random numbers with a method based on Sobol sequences. Sobol sequences belong to the family of quasi-random sequences which are designed to generate samples of multiple parameters as uniformly as possible over the multi-dimensional parameter space (Saltelli et al., 2010). The biggest difference to

¹Matlab is a trademark of The MathWorks Inc.

pseudo-random numbers is that the sample values are chosen under consideration of the previously sampled points and thus avoiding the occurrence of clusters and gaps (Burhenne et al., 2011).

In practical terms, first we use Sobol sequence to generate the space filling design on $[0, 1]^D$, where D is the dimension of each of the design points \mathbf{x}_i , $i = 1, \dots, d$. Then, those design points are pushed through the inverse cumulative distribution function of a $\mathcal{N}(0, \sigma_d^2)$ random variable ($\sigma_d^2 > 1$). We use $\sigma_d^2 > 1$ to ensure that the design points are slightly more spread out than the random variables ξ_i (which have a $\mathcal{N}(0, 1)$ distribution) and thus we assure that we cover the tiles of the distribution we want to model. Once a design is chosen, we need to run the simulator once for each design point in order to built our training set,

$$\mathcal{D} = \{(\mathbf{x}_i, y_i = f(\mathbf{x}_i)), i = 1, \dots, d\}. \quad (2.1)$$

The election of the covariance function is crucial in a GP predictor, as it encodes our assumptions about the function which we wish to learn (Rasmussen and Williams, 2006). It is the covariance function that defines the concept of *nearness*, i.e. two points x_1 and x_2 which are close in distance are likely to have close target values $y_1 = f(x_1)$ and $y_2 = f(x_2)$, and so arbitrary functions will not, in general, be a valid covariance function for our model. Throughout this section we will present examples of some commonly used covariance functions with the purpose of analyzing and testing the performance of our GP emulator. In order for a model to be a practical tool, we needs to make decisions about the details of its specification. Some properties may be easy to specify from the context of the problem, while we typically have only vague information available about other aspects, e.g., length-scales or process variances. In order to learn suitable functional forms for our covariance functions we express them explicitly in terms of some continuous parameters, which we call *hyperparameters*, and so any given covariance function will have a number of hyperparameters whose values also need to be determined (or inferred). The process of learning suitable values for any given set of hyperparameters is called *to train the model* (see section 2.5 for details about how to train the model).

2.2 The prior covariance function

For our GP emulator we have chosen three different families of covariance functions. The first one is called the squared exponential (*SE*) covariance function. This covariance function is infinitely differentiable, which means that the GP with this covariance function has mean square derivatives¹ of all orders, and is thus very smooth (MS differentiability). Although the SE is probably the most widely used covariance function, Stein (1999) argues that this strong smoothness assumptions is unrealistic for modelling many physical processes, and recommends the Matérn class (Matérn, 1960), used also by Cornford et al. (2002), and so, that will be our second choice². The third covariance function, chosen to give alternative view from the previous two choices, will be the rational quadratic (RQ) which was presented as an alternative to the Matérn class by Matérn (1960). The expressions for the squared-exponential (2.2), Matérn class (2.3) and (2.4), and rational quadratic (2.5) covariance functions in one dimension in terms of their corresponding hyperparameters have the following form:

$$k_{SE}(x_i, x_j) = \sigma_f^2 \exp\left(-\frac{|x_i - x_j|^2}{2\ell}\right) + \sigma_n^2 \mathbf{I}, \quad (2.2)$$

$$k_{\frac{3}{2}}(x_i, x_j) = \sigma_f^2 \left(1 + \frac{\sqrt{3}|x_i - x_j|}{\ell}\right) \exp\left(-\frac{\sqrt{3}|x_i - x_j|}{\ell}\right) + \sigma_n^2 \mathbf{I}, \quad (2.3)$$

$$k_{\frac{5}{2}}(x_i, x_j) = \sigma_f^2 \left(1 + \frac{\sqrt{5}|x_i - x_j|}{\ell} + \frac{5|x_i - x_j|^2}{3\ell^2}\right) \exp\left(-\frac{\sqrt{5}|x_i - x_j|}{\ell}\right) + \sigma_n^2 \mathbf{I}, \quad (2.4)$$

¹Following section 4.1.1 in Rasmussen and Williams (2006) we describe mean square continuity and differentiability of stochastic process (like GP) as follows: Let $\mathbf{x}_1, \mathbf{x}_2, \dots$ a sequence of points and \mathbf{x}' be a fixed point in \mathbb{R}^n , $n \in \mathbb{N}$, such that $|\mathbf{x}_k - \mathbf{x}'| \rightarrow 0$ as $k \rightarrow \infty$. Then a process $f(x)$ is continuous in mean square (MS) at \mathbf{x}' if $\mathbb{E}[|f(\mathbf{x}_k) - f(\mathbf{x}')|^2] \rightarrow 0$ as $k \rightarrow \infty$. The mean square derivative of $f(x)$ in the i th direction is then defined as $\frac{\partial f(\mathbf{x})}{\partial x_i} = \lim_{h \rightarrow 0} \frac{f(\mathbf{x} + h\mathbf{e}_i) - f(\mathbf{x})}{h}$, when this limit exists , where this limits denotes limit in mean square and \mathbf{e}_i is the unit vector in the i th direction.

²There are a whole family of Matérn class functions and in this work we will use only two representatives, the Matérn _{$\frac{3}{2}$} and Matérn _{$\frac{5}{2}$} .

$$k_{RQ}(x_i, x_j) = \sigma_f^2 \left(1 + \frac{|x_i - x_j|^2}{2\alpha\ell^2} \right)^{-\alpha} + \sigma_n^2 \mathbf{I}, \quad (2.5)$$

where ℓ is the characteristic length-scale, σ_f^2 is the variance of the process, $\alpha > 0$, σ_n^2 is the variance of the induced Gaussian noise.

For simplicity, we have presented only one dimensional covariance functions, but in general, a multi-dimensional treatment has to be given to the covariance functions in order to obtain a realistic model. This is done by introducing multi-dimensional characteristic length scales in the form $\boldsymbol{\ell} = (\ell_1, \dots, \ell_D)$, where D is the dimension of the inputs for our model (the dimension of the training points). The idea is, roughly, how far do you need to move (along a particular axis) in input space for the function values to become uncorrelated. Such a covariance function implements automatic relevance determination (ARD) (Neal, 1996) since the inverse of the length-scale determines how relevant an input is: if the length-scale has a very large value, the covariance will become almost independent of that input, effectively removing it from the inference (Rasmussen and Williams, 2006).

Anisotropic versions of these covariance functions can be created by setting $(\mathbf{x}_i - \mathbf{x}_j)^2 = (\mathbf{x}_i - \mathbf{x}_j)^\top \mathbf{M}(\mathbf{x}_i - \mathbf{x}_j)$ in (2.2), (2.3), (2.4) and (2.5), for some positive semi-definite matrix \mathbf{M} . If we choose \mathbf{M} to be diagonal, this will implement the use of different length-scales on different dimensions. So, we will set $\mathbf{M} = \text{diag}(\frac{1}{\ell_i^2})$, with $i = 1, \dots, D$. So, anisotropic versions of the covariance functions parameterized in terms of hyperparameters are explicitly given by,

$$k_{SEard}(\mathbf{x}_i, \mathbf{x}_j) = \sigma_f^2 \exp \left(-\frac{1}{2} (\mathbf{x}_i - \mathbf{x}_j)^\top \mathbf{M} (\mathbf{x}_i - \mathbf{x}_j) \right) + \sigma_n^2 \delta_{ij}, \quad (2.6)$$

$$k_{\frac{3}{2}ard}(\mathbf{x}_i, \mathbf{x}_j) = \sigma_f^2 s_{\frac{3}{2}}(r_{\frac{3}{2}}(\mathbf{x}_i, \mathbf{x}_j)) \exp(-r_{\frac{3}{2}}(\mathbf{x}_i, \mathbf{x}_j)) + \sigma_n^2 \delta_{ij}, \quad (2.7)$$

$$k_{\frac{5}{2}ard}(\mathbf{x}_i, \mathbf{x}_j) = \sigma_f^2 s_{\frac{5}{2}}(r_{\frac{5}{2}}(\mathbf{x}_i, \mathbf{x}_j)) \exp(-r_{\frac{5}{2}}(\mathbf{x}_i, \mathbf{x}_j)) + \sigma_n^2 \delta_{ij}, \quad (2.8)$$

with

$$s_{\frac{3}{2}}(t) = 1 + t, \quad s_{\frac{5}{2}}(t) = 1 + t + \frac{t^2}{3}$$

and

$$r_{\frac{3}{2}}(\mathbf{x}_i, \mathbf{x}_j) = (3(\mathbf{x}_i - \mathbf{x}_j)^\top \mathbf{M}(\mathbf{x}_i - \mathbf{x}_j))^{\frac{1}{2}}, \quad r_{\frac{5}{2}}(\mathbf{x}_i, \mathbf{x}_j) = (5(\mathbf{x}_i - \mathbf{x}_j)^\top \mathbf{M}(\mathbf{x}_i - \mathbf{x}_j))^{\frac{1}{2}},$$

$$k_{RQard}(\mathbf{x}_i, \mathbf{x}_j) = \sigma_f^2 \left(1 + \frac{1}{2\alpha} (\mathbf{x}_i - \mathbf{x}_j)^\top \mathbf{M}(\mathbf{x}_i - \mathbf{x}_j) \right)^{-\alpha} + \sigma_n^2 \delta_{ij}, \quad (2.9)$$

where δ_{ij} is the Kronecker delta. Explicitly, the corresponding hyperparameters for each case are, respectively,

$$\boldsymbol{\theta}_{SE} = (\sigma_f^2, \boldsymbol{\ell}, \sigma_n^2), \quad (2.10)$$

$$\boldsymbol{\theta}_{Matern} = (\sigma_f^2, \boldsymbol{\ell}, \sigma_n^2), \quad (2.11)$$

$$\boldsymbol{\theta}_{RQ} = (\sigma_f^2, \boldsymbol{\ell}, \alpha, \sigma_n^2). \quad (2.12)$$

2.3 The prior mean function

It is common but by no means necessary to consider GPs with a zero mean function. This is not necessarily a drastic limitation, since the mean of the posterior process is not confined to be zero (Rasmussen and Williams, 2006). Constant and linear mean functions have been tested for the models studied in this thesis and proved to give similar results to a zero mean, while the computational cost was bigger. Thus, in this thesis we will only show the results obtained by using a zero mean function in all our GP models.

2.4 The likelihood function

For our Bayesian regression model we will consider the likelihood function as Gaussian. This election is due to the fact that a Gaussian process prior combined

with a Gaussian likelihood gives rise to a posterior Gaussian process over functions (Rasmussen and Williams, 2006), and thus, all the process remains analytically tractable for our purpose.

2.5 Training our Gaussian process model

Once we have the prior functions in terms of their hyperparameters we want to be able to make inferences about all of them in the light of the data. In order to do this we compute the probability of the data given the hyperparameters, i.e.,

$$p(\mathbf{y}|X) = \int p(\mathbf{y}|X, \boldsymbol{\theta})p(\boldsymbol{\theta})d\boldsymbol{\theta}. \quad (2.13)$$

We can now find the values of the hyperparameters which optimizes the marginal likelihood (ML) (2.13) or equivalently, find the values of the hyperparameters which optimizes the *negative log marginal likelihood* (*NLML*). As we let $\mathbf{y} \sim \mathcal{N}(\mathbf{0}, \Sigma + \sigma_n^2 \mathbf{I})$ (see 1.19), the negative log marginal likelihood can be expressed by (Rasmussen and Williams, 2006),

$$NLML = -\log p(\mathbf{y}|X) = -\frac{1}{2}\mathbf{y}^\top(\Sigma + \sigma_n^2 \mathbf{I})^{-1}\mathbf{y} - \frac{1}{2}\log|\Sigma + \sigma_n^2 \mathbf{I}| - \frac{n}{2}\log 2\pi. \quad (2.14)$$

For doing this we will use a routine implemented within the code GPML which approximates the optima values using a local expansion around the maximum (the Laplace approximation) (see Rasmussen and Williams (2006), Section 5.2 for details). Once we have obtained estimates of the hyperparameters, we can use the expressions established for prediction (1.22) and (1.23).

So far we have introduced all necessary material to build a GP emulator, but sometimes (most of the cases for real-world problems), we do not have access to either the analytical solution of a mathematical problem or experimental data to validate our model. In this case, in order to be confidence on the performance of the GP emulator, we test our emulator by comparing our predictions with the observed values available in our training set. We will refer to this process

as *cross-validation* (CV) and we will discuss how to use CV for testing our GP emulator in the following section.

2.6 Cross-validation

In this section we consider how to use methods of CV for model testing and selection. The basic idea is to split the training set into two disjoint sets, one of which is used for training, and the other, the validation set, is used to monitor performance. The performance on the validation set is used to estimate the prediction error and model testing and selection are carried out using this measure. We will use an extreme case of the k-fold cross-validation known as leave-one-out cross-validation (LOO-CV). During LOO-CV, we fit a model using all but a single data point, and then we compute the model prediction error on the left out point. We then repeat this for all points. Cross-validation can be used with any loss function¹ although the squared error loss is the most common for regression, for probabilistic models such as GP it is natural to consider also cross-validation using the negative log probability loss (Rasmussen and Williams, 2006). Another possibility (Wilkinson et al., 2011) is to use the Dawid score introduced by Dawid and Sebastiani (1999). To compute those quantities we have followed the formulae proposed in Wilkinson et al. (2011). Suppose that we call m_j our predicted expected value, and s_j^2 its corresponding variance, and we call \hat{y}_j our observed (true) value, then, we define the mean squared error (MSE) and Dawid score (DS) as,

$$MSE = \frac{1}{N} \sum_{j=1}^N (\hat{y}_j - m_j)^2, \quad (2.15)$$

$$DS = \frac{1}{N} \sum_{j=1}^N \left(\frac{(\hat{y}_j - m_j)^2}{s_j^2} + \log s_j^2 \right). \quad (2.16)$$

We will show how we use LOO-CV to test our particular GP model and to select adequate model specifications in Section 2.9.1. Let us put all these concepts

¹A loss function is a function which specifies the loss or penalty incurred by guessing a value that does not correspond to the true value (Rasmussen and Williams, 2006).

and tools in practice through an application example. In this application we test the GP emulator with a prior zero-mean function and the four covariance functions introduced in Section 2.2, namely, *SE* (2.6), *Matérn_{3/2}* (2.7), *Matérn_{5/2}* (2.8) and *RQ* (2.9). The properties of the Sobol sequences require a sample size of 2^j where $j \in \mathbb{N}$ (Burhenne et al., 2011) in order to exploit the way that the points are spread when filling the space, and thus, the number of design points for each training set to be considered throughout this thesis will be 256 (2^8) points.

2.7 Travel time of a convected particle in groundwater flow.

Modelling adequately groundwater systems is essential for the safety of geological disposal facilities, for instance, the leakage of radionuclides which are transported through groundwater flow and which has direct impact on water resources. Nowadays, computer simulations have become an indispensable modeling tool for the groundwater research. Normally, the simulation of groundwater flow involves intrinsic uncertainties due to the lack of knowledge of the porous medium where the particles are flowing through. Thus, this uncertainty propagates throughout the calculations, and quantification of its impact on results of computer simulations is the core issue of ongoing research in this field.

In the following, we introduce the physical problem we wish to study, the quantities of interest, and provide details of the mathematical model we propose to quantify the uncertainty in those quantities of interest.

2.8 Mathematical model

The study of groundwater flow is well established and there is general scientific consensus that in many situations Darcy's law can be expected to lead to an accurate description of the flow (Cliffe et al., 2011). The main parameter appearing in Darcy's law is the hydraulic conductivity, which characterises how

easily the fluid can flow through the rock under a given pressure gradient.

Stone (2011) gives a detailed description of the parameters involved in groundwater flow models. We will introduce here only the relevant parameters for our model. These parameters are, the porosity, ϕ (dimensionless), which is the ratio of void volume in a rock to total volume, it indicates how much fluid a rock can hold, the permeability (of a rock), K (m^2), which measures the ability of the rock to transmit fluid through its pores. The driving force behind the movement of groundwater is head gradient. Head, h (m), is the height, above an arbitrary given level, that a fluid in a rock can reach due to the fluid pressure, P ($\text{kg m}^{-1} \text{s}^{-2}$). The density of the fluid is denoted by ρ (kg m^{-3}). The head gradient is the difference in head between two points in a region over the distance between those two points. Note that the fluid will always flow from areas of high head to areas of low head. \mathbf{q} (m s^{-1}), is the filtration velocity (or Darcy's flux), and b (m) is the thickness of a layer of rock.

2.8.1 Governing equations

Noting the given notation for the parameters involved in our model, the classical equations governing (steady-state) single phase subsurface flow consist of Darcy's law (2.17) coupled with an incompressibility condition (2.18) (see e.g., de Marsily (1986); Cliffe et al. (2000b); Stone (2011)), i.e.,:

$$\mathbf{q} + K \nabla P = \mathbf{g}_s, \quad (2.17)$$

$$\nabla \cdot \mathbf{q} = 0, \quad (2.18)$$

where \mathbf{g}_s are the source terms. These equations are subject to the following boundary conditions:

2.8.2 Domain and boundary conditions

The convection process is considered to occur in a horizontal section of porous material represented by a cartesian system with domain $D = [0, L] \times [0, H] \subset \mathbb{R}^2$.

The boundary conditions are:

Dirichlet:

$$P(x = 0, y) = P_{in}, \quad P(x = L, y) = P_{out}, \quad (2.19)$$

where P_{in} and P_{out} denote the pressure at left boundary and right boundary respectively.

Neumann:

$$\frac{\partial P}{\partial y}(x, y = 0) = 0, \quad \frac{\partial P}{\partial y}(x, y = H) = 0. \quad (2.20)$$

In the next section we consider an example of a particle convected in the fluid, passing through a horizontal layer of porous rock confined from above and below as stated in (2.20). The quantity of interest will be the averaged travel time that the convected particles released from the middle of the rock last to reach the end of the porous rock.

2.8.3 Quantity of interest

We are interested in estimating the averaged travel time of convected particles in groundwater flow. For doing this, the equations (2.17) and (2.18) can be coupled and solved for the pressure. After the pressure is calculated, the travel time that a convected particle initially located at the center, $(x_0 = \frac{L}{2}, y_0 = \frac{H}{2})$, of the domain, D , last to reach the right boundary, $\mathbf{z}(\tau) = (L, y)$, can be computed by integrating the transport equation (Stone (2011); Bear (1972)),

$$\frac{d\mathbf{z}(t)}{dt} = \frac{q(x)}{b\phi} = -\frac{K(x)}{b\phi} \nabla P(x), \quad (2.21)$$

i.e.,

$$\mathbf{z}(t) = \mathbf{z}(0) + \int_0^t -\frac{K(\mathbf{z}(\zeta))}{b\phi} \nabla P(\mathbf{z}(\zeta)) d\zeta \quad (2.22)$$

and therefore we can compute the time, τ , for which $x(\tau) = L$, i.e.,

$$L = \mathbf{z}(\tau) = \mathbf{z}(0) + \int_0^\tau -\frac{K(\mathbf{z}(\zeta))}{b\phi} \nabla P(\mathbf{z}(\zeta)) d\zeta. \quad (2.23)$$

To solve the equation (2.23) we use a simulator¹ which receives as input the permeability and returns as scalar output the corresponding travel time, τ .

To test the precision of the simulator we solve an example problem for which we know that there is an analytical solution and thus we can obtain a direct comparison. The problem to solve is:

$$\nabla^2 P = 0, \quad (x, y) \in D = [0, 1] \times [0, 1], \quad (2.24)$$

with boundary conditions,

$$P(x, 0) = P(x, 1) = 10(1 - x), \quad (2.25)$$

$$P(0, y) = 10,$$

$$P(1, y) = 0.$$

Note that for simplicity we have considered $k = 1$ in D . This problem has the analytical solution $P(x, y) = 10 - 10x$. To compute the travel time, we consider a particle released at the centre, $\mathbf{z} = (0.5, 0.5)$, of the domain D and solve

$$\frac{d\mathbf{z}(t)}{dt} = -\nabla P(x), \quad (x, y) \in [0, 1] \times [0, 1], \quad (2.26)$$

with

$$\mathbf{z}(0) = (0.5, 0.5).$$

¹This simulator, implemented in Matlab, was developed by Dr. Marco Iglesias (School of Mathematics, University of Nottingham) and is based on the standard cell-centred finite volume method described in detail in Cliffe et al. (2011).

Thus, the travel time can be derived from the following equation:

$$\frac{d\mathbf{z}(t)}{dt} = -\nabla(10 - 10x) = (10, 0), \quad (2.27)$$

which by integration and application of the initial condition, $\mathbf{z}(0) = (0.5, 0.5)$ leads to the explicit solution

$$\mathbf{z}(t) = (0.5 + 10t, 0.5). \quad (2.28)$$

As the velocity of the particle in the x-direction, $\frac{\partial \mathbf{z}}{\partial x}$, is positive, we can anticipate that the particle will leave the region by the east boundary, and thus, by solving $1 = 0.5 + 10t$ we obtain $t = 0.05$.

The numerical solution given by our simulator was exactly the same $t = 0.05$ for this example problem. The simulation of the corresponding trajectory is shown in Figure 2.1.

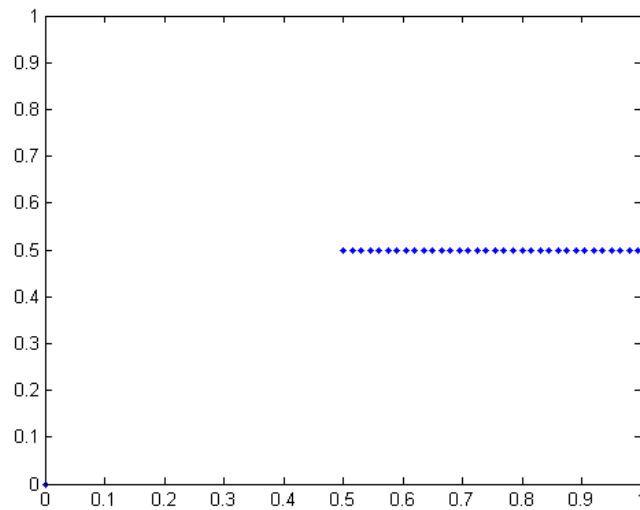


Figure 2.1: Simulated trajectory of the convected particle initially released at the center of the domain for the problem (2.24) with the boundary conditions given in (3.1).

In the previous example we have considered the parameter K to be constant in all the domain, but this is not an accurate representation of the reality. Bear

(1972) provides some empirical data for the permeability and classify several scenarios (Figure 2.2). Farthing et al. (2012) shows some results obtained when dealing with different porous media scenarios by using different sort of sands. It has been shown (Byers and Stephens (1983); Hoeksema and Kitanidis (1985); Russo and Bouton (1992)) that although the permeability values can show large spatial variations, these variations are not entirely random but spatially correlated, and, these porous media samples have been previously modelled as log-normal fields (Mondal et al., 2010).

Throughout this thesis we will use a log-normal distribution to model the parameter K , i.e., in the two dimensional case we replace the conductivity tensor by a scalar valued field whose log is Gaussian. In the next section we describe how we model the permeability as a log normal random field and how the model presented in Section 2.8.1 yields to a system of PDEs with random coefficients.

Table 5.5.1
Typical Values of Hydraulic Conductivity and Permeability

$-\log_{10} \cdot K(\text{cm/s})$	-2	-1	0	1	2	3	4	5	6	7	8	9	10	11	
Permeability	Pervious			Semipervious			Impervious								
Aquifer	Good			Poor			None								
Soils	Clean gravel	Clean sand or sand and gravel		Very fine sand, silt, loess, loam, solonetz											
				Peat		Stratified clay		Unweathered clay							
Rocks				Oil rocks			Sandstone		Good limestone, dolomite	Breccia, granite					
$-\log_{10} \cdot k(\text{cm}^2)$	3	4	5	6	7	8	9	10	11	12	13	14	15	16	
$\log_{10} k(\text{md})$	8	7	6	5	4	3	2	1	0	-1	-2	-3	-4	-5	

Figure 2.2: Typical empirical values of hydraulic conductivity and permeability.
(Image extracted from Bear (1972)).

2.8.4 Generation of random permeability fields

Let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space. We will call random field (RF) and write $\{Z(\mathbf{x}) : \mathbf{x} \in D\}$ or just $Z(\mathbf{x}, \omega)$ to a set of real-valued random variables on the probability space $(\Omega, \mathcal{F}, \mathbb{P})$. So, for each $\mathbf{x} \in D$, $Z(\mathbf{x}, \cdot) : \Omega \rightarrow \mathbb{R}$ is a random variable. For a given RF, $Z(\mathbf{x}, \omega)$, we define the mean function $m : D \rightarrow \mathbb{R}$ by:

$$m(\mathbf{x}) = \mathbb{E}[Z(\mathbf{x}, \omega)] = \int_{\Omega} Z(\mathbf{x}, \omega) d\mathbb{P}(\omega), \quad \mathbf{x} \in D,$$

and the covariance function $c : D \times D \rightarrow \mathbb{R}$, by:

$$c(\mathbf{x}, \mathbf{x}') = \mathbb{E}[(Z(\mathbf{x}, \omega) - m(\mathbf{x}))(Z(\mathbf{x}', \omega) - m(\mathbf{x}'))], \quad \mathbf{x}, \mathbf{x}' \in D. \quad (2.29)$$

For a fixed $\omega \in \Omega$, we will call the associated *realisation* of the RF, $Z(\mathbf{x}, \omega)$, to the deterministic function $Z(\cdot, \omega) : D \rightarrow \mathbb{R}$. Taking one realisation of each of the random variables $Z(\mathbf{x})$, we get a discretised function of \mathbf{x} . This realisation then gives a possible representation of the true field for use in a computer model. The simulator described in Section 2.8.3 only requires as inputs the values of the permeability at certain locations of the domain, D , rendering the initial continuous domain to a finite computational domain. We will refer to these locations at which the permeability is evaluated as *nodes*, and we will denote the set of nodes by Ω_M , where M denotes the number of nodes required by the simulator in order to return an output. If we now let $Z(\mathbf{x})$ be a Gaussian RF, we can take $\mathbf{K} = \exp(Z(\mathbf{x}))$ which is then a log Gaussian random field (Lord et al., 2014). In this case, our simulator will receive as inputs a log Gaussian random field, \mathbf{K} , and will return a scalar output, the corresponding travel time, τ . Henceforth, we can refer to the simulator as a function, f , as follows:

$$f : \mathbf{K} \in \mathbb{R}^M \rightarrow \tau \in \mathbb{R}. \quad (2.30)$$

One possibility to generate a Gaussian RF, Z , is to use the matrix decomposition of the covariance matrix associated to the covariance function given in

(2.29), such as Cholesky¹ decomposition. One inconvenience is that even for a few hundreds of the sampling points the round-off error is very significant due to the fact that the associated covariance matrix is likely to become extremely ill-conditioned Dietrich and Newsam (1989). Other alternative method for simulating Gaussian RF is the circular embedding algorithm (Dietrich and Newsam, 1997) described in Lord et al. (2014). This method provides an exact simulation of Gaussian RF, but it requires extensive use of the theory of the functional analysis to be implemented. In this thesis, we will opt for the KL decomposition method (Ghanem and Spanos, 1991), described next in Section 2.8.5, to generate the random permeability fields. This method could be inappropriate for problems where the simulator necessitates of a very fine discretization of the computational domain in order to provide accurate outputs, but this does not apply to our simulator. Conversely, the advantage of this approach is that it only requires one eigen-decomposition of the covariance matrix associated to the covariance function given in (2.29) and then that structure is stored and used to generate fast new realisations of the permeability field.

2.8.5 KL decomposition

Let $Z(\mathbf{x}, \omega)$ be a RF with mean $m(\mathbf{x})$ and covariance $c(\mathbf{x}, \mathbf{x}')$ associated with $(\Omega, \mathcal{F}, \mathbb{P})$. Given the set of points $\{\mathbf{x}_i \in D : i = 1, \dots, M\}$, the vector $\mathbf{Z} := [Z(\mathbf{x}_1, \omega), \dots, Z(\mathbf{x}_M, \omega)]^\top$ is a discrete random field. In fact, $\mathbf{Z} : \Omega \rightarrow \mathbb{R}^M$ is a multivariate random variable with mean vector $\mathbf{m} = \mathbb{E}[\mathbf{Z}] \in \mathbb{R}^M$ and covariance matrix $\mathbf{C} = \mathbb{E}[(\mathbf{Z} - \mathbf{m})(\mathbf{Z} - \mathbf{m})^\top] \in \mathbb{R}^{M \times M}$, where, for $i, j = 1, \dots, M$, $m_i := \mathbb{E}[Z(\mathbf{x}_i, \omega)] = m(\mathbf{x}_i)$, and $C_{ij} := c(\mathbf{x}_i, \mathbf{x}_j)$. We now can take $\mathbf{K} := \exp(\mathbf{Z})$ as a discrete log Gaussian permeability field representing our parameter k .

In order to generate realisations of \mathbf{K} which can be introduced as inputs in our simulator we need to first generate samples of $\mathbf{Z} \sim \mathcal{N}(\mathbf{m}, \mathbf{C})$. A covariance function that has been extensively used in the literature (e.g., Hoeksema and Kitanidis (1985); Stone (2011); Collier et al. (2014); Cliffe et al. (2011)) for

¹The Cholesky decomposition of a Hermitian positive-definite matrix, \mathbf{A} , is a decomposition of the form, $\mathbf{A} = \mathbf{L}\mathbf{L}^*$, where \mathbf{L} is a lower triangular matrix with real and positive diagonal entries, and \mathbf{L}^* denotes the conjugate transpose of \mathbf{L} (see e.g., Strang (2003)).

modelling the correlation length scale for k is the following exponential two-point covariance function

$$c(\mathbf{x}_i, \mathbf{x}_j) = \sigma^2 \exp\left(\frac{-\|\mathbf{x}_i - \mathbf{x}_j\|_2}{\lambda}\right) \quad \mathbf{x}_i, \mathbf{x}_j \in D, \quad (2.31)$$

where $\|\cdot\|_2$ represent the ℓ_2 -norm in \mathbb{R}^2 , the parameter λ represents the correlation length and σ^2 represents the variance, and in subsurface flow applications typically only $\sigma \geq 1$ and $\lambda \leq \text{diam } D = 1$ will be of interest (Cliffe et al., 2011). We denote by \mathbf{C} to the positive semi-definite covariance matrix associated to the function c , i.e. $C_{ij} = c(\mathbf{x}_i, \mathbf{x}_j)$. As this covariance matrix \mathbf{C} is real-valued and symmetric, by the eigen-decomposition theorem (see e.g., Strang (2003)) we can express \mathbf{C} by:

$$\mathbf{C} = \Phi \Lambda \Phi^\top, \quad \text{or,} \quad \mathbf{C} = \Phi \Lambda^{\frac{1}{2}} \Lambda^{\frac{1}{2}} \Phi^\top = (\Phi \Lambda^{\frac{1}{2}})(\Phi \Lambda^{\frac{1}{2}})^\top,$$

where Λ is the $M \times M$ diagonal matrix of ordered decreasing eigenvalues $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_M \geq 0$, and Φ is the $M \times M$ matrix whose columns ϕ_i $i = 1, \dots, M$, are the eigenvectors of C .

If we now let $\xi = [\xi_1, \dots, \xi_M]^\top \sim \mathcal{N}(\mathbf{0}, I)$, and consider $\mathbf{Z} = \mathbf{m} + \Phi \Lambda^{\frac{1}{2}} \xi$, we can write (Lord et al., 2014),

$$\mathbf{Z} = \mathbf{m} + \Phi \Lambda^{\frac{1}{2}} \xi = \mathbf{m} + \sum_{i=1}^M \sqrt{\lambda_i} \phi_i \xi_i, \quad \xi_i \sim \mathcal{N}(0, 1) \quad i.i.d.,$$

and consequently the discrete log-Gaussian random field,

$$\mathbf{K} = \exp\left(\mathbf{m} + \sum_{i=1}^M \sqrt{\lambda_i} \phi_i \xi_i\right). \quad (2.32)$$

The random variables ξ_i will be called *KL coefficients*. Sometimes and depending on the computational resources, the number M can be very large. So we can use the mean-square error (MSE) to approximate our discrete RF with a desired level of accuracy or tolerance.

If we let $\hat{\mathbf{Z}} = \mathbf{m} + \sum_{i=1}^{H_M} \sqrt{\lambda_i} \phi_i \xi_i$ be the truncated sum of \mathbf{Z} after H_M terms, we can compute the MSE between \mathbf{Z} and $\hat{\mathbf{Z}}$ as follows (see Lord et al. (2014) for

details),

$$\begin{aligned}\mathbb{E} \left[\|\mathbf{Z} - \hat{\mathbf{Z}}\|_2^2 \right] &= \mathbb{E} \left[\sum_{i=H+1}^M \sum_{j=H+1}^M \sqrt{\lambda_i} \sqrt{\lambda_j} \phi_i \phi_j^\top \xi_i \xi_j \right] \\ &= \sum_{i=H+1}^M \sum_{j=H+1}^M \sqrt{\lambda_i} \sqrt{\lambda_j} \phi_i \phi_j^\top \mathbb{E} [\xi_i \xi_j] = \sum_{j=H+1}^M \lambda_j.\end{aligned}\tag{2.33}$$

Note that we have used the linearity of the expectation and the orthogonality of the eigenvectors in (2.33), i.e., $\phi_i \phi_j^\top = 0$ for $i = j$, and $\phi_i \phi_j^\top = 0$ for $i \neq j$. Now we can calculate the relative error of approximating \mathbf{Z} with $\hat{\mathbf{Z}}$, $E = \frac{\mathbb{E}[\|\mathbf{Z} - \hat{\mathbf{Z}}\|_2^2]}{\mathbb{E}[\|\mathbf{Z}\|_2^2]}$, and choose a desired tolerance, $\beta_{tol} \in (0, 1)$, to compute the corresponding number of terms H_M for the approximation, i.e.,

$$0 \leq E \leq \beta_{tol} < 1.\tag{2.34}$$

Note that $E = 0$ when $H_M = M$.

Once modelled the permeability parameter, K , as a log Gaussian random field, $K = K(\mathbf{x}, \omega)$ on $D \times \Omega$, the equations (2.17) and (2.18) become a system of PDEs with random coefficients (Cliffe et al., 2011), which can be written in second order form as

$$-\nabla \cdot (K(\mathbf{x}, \omega) \nabla P(\mathbf{x}, \omega)) = h_s(\mathbf{x}), \quad \text{in } D,\tag{2.35}$$

with $h_s := -\nabla \cdot \mathbf{g}_s$.

In the following, we will focus on the problem given by equation (2.35) and the boundary conditions,

$$P(0, y) = 100, \quad P(1, y) = 0,$$

$$\frac{\partial P}{\partial y}(x, 0) = 0,$$

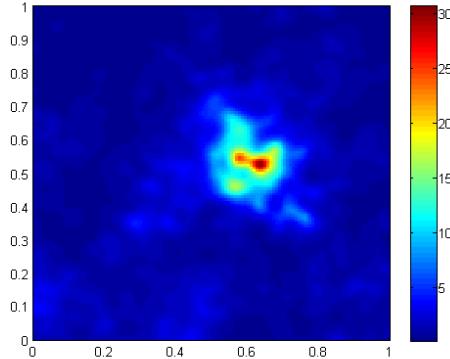
$$\frac{\partial P}{\partial y}(x, 1) = 0,$$

where the sources \mathbf{g}_s are known (and thus deterministic), and restrict ourselves to the case $D = [0, 1] \times [0, 1] \subset \mathbb{R}^2$ (i.e., $L = H = 1$) and $h \equiv 0$.

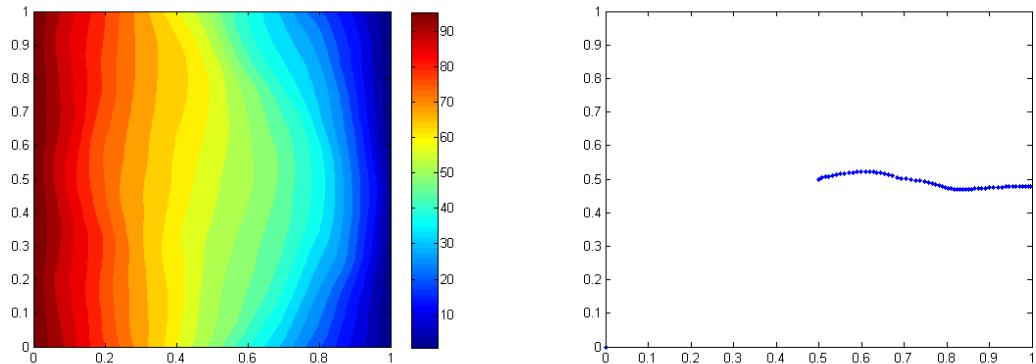
A single random permeability field, \mathbf{K} , could represent possible sets of permeabilities values in a slice of rock, across which we would like to study a fluid flow. In this case, we will refer to them as *heterogeneous* permeability field. Note, for instance, that in the example (2.24) we assumed that the permeability field was *homogeneous* by setting $K = 1$ in D . We will study this concept of heterogeneity further in Chapter 4.

Examples of heterogeneous permeability fields, pressure fields and trajectories for convected particles obtained by solving the problem (2.35) are showed in Figures 2.3 and 2.4.

In the following section we will show how we build the GP emulator and how we use it to make predictions.



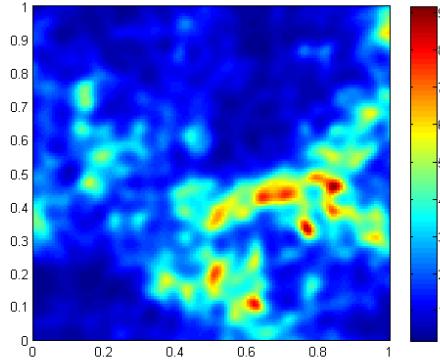
(a) Simulated heterogeneous permeability field in domain D .



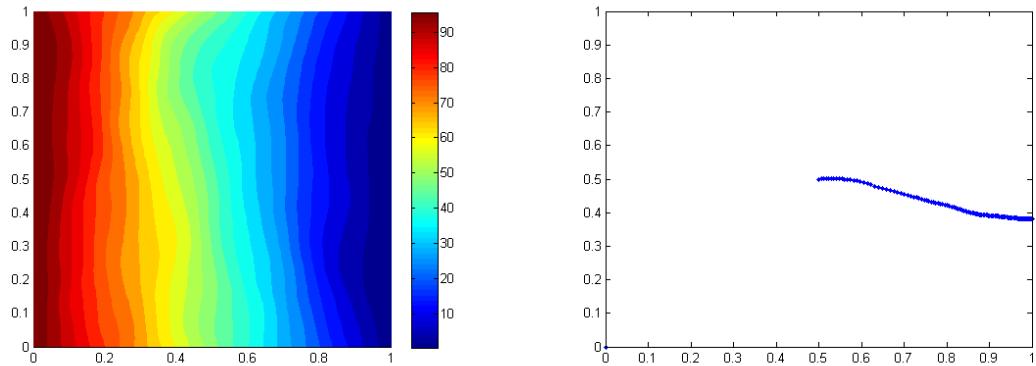
(b) Simulated pressure field contours for the given permeability.

(c) Simulated trajectory of a convected particle released at the center of the domain.

Figure 2.3: Example of a heterogeneous permeability field (top) generated with the KL decomposition method described in Section 2.8.5. Simulated pressure field (bottom left) for the given permeability from equation (2.35). Simulated trajectory (bottom right) of a convected particle released at the center of the domain. The travel time spent for the particle in reaching the right boundary is 0.59 seconds.



(a) Simulated heterogeneous permeability field in domain D .



(b) Simulated pressure field contours for the given permeability. (c) Simulated trajectory of a convected particle released at the center of the domain.

Figure 2.4: Example of heterogeneous permeability field (top) generated with the KL decomposition method described in Section 2.8.5. Simulated pressure field (bottom left) for the given permeability from equation (2.35). Simulated trajectory (bottom right) of a convected particle released at the center of the domain. The travel time spent for the particle in reaching the right boundary is 1.09 seconds.

2.9 Using the GP emulator for making predictions

Let $f(\cdot)$ represent the simulator described in (2.30), which takes as input a random permeability field and returns the corresponding travel time, τ , as discussed in the previous section. A general methodology for building a GP emulator for the simulator $f(\cdot)$, as illustrated in Figure 2.5, can be summarised as:

1. Choose the appropriate set of design points, ξ_i , where to run the simulator.
2. Form the corresponding random permeability fields, \mathbf{K}_i , associated with the design points ξ_i .
3. Run the simulator, $f(\cdot)$, at those \mathbf{K}_i and obtain the corresponding τ_i in order to form the training set \mathcal{D} .
4. Use GP regression described in Section 1.3.3 to approximate the entire function $f(\cdot)$ with $\hat{f}(\cdot)$.

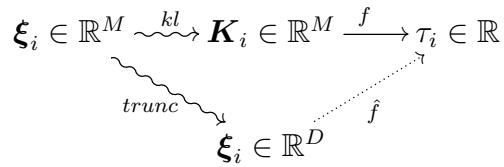


Figure 2.5: Emulation diagram. Above, the set of design points, ξ_i , used to generate the log Gaussian RF, \mathbf{K}_i , with a KL decomposition. These \mathbf{K}_i are then used to compute the corresponding travel times τ_i with the simulator f . These τ_i are then used as observed values in the training set. Below, the set of truncated design points, $\xi_i = (\xi_1, \xi_2, \dots, \xi_D)$, to form the training set along with the observed travel times, τ_i . \hat{f} represents the GP emulator which is able to predict the travel time, $\tau_i^* \in \mathbb{R}$, for a given test case $\xi_i^* \in \mathbb{R}^D$.

In the following section we will use the LOO-CV methodology described in Section 2.6 to help us to decide which of the covariance functions, SE (2.6),

$\text{Matérn}_{\frac{3}{2}}$ (2.7), $\text{Matérn}_{\frac{5}{2}}$ (2.8) or RQ (2.9) would be appropriate to approximate our simulator.

2.9.1 GP emulator test and selection

To select the GP model that more accurately approximates our emulator, $f(\cdot)$, we apply the LOO-CV method to a training set of 256 points. For a well calibrated GP emulator, the data will lie within the 95% CI 95% of the time (Henderson et al., 2009). Figures 2.6, 2.7, 2.8 and 2.9 show the predicted values (red) and 95% confidence intervals (black vertical bars) for each of the predictions, along with the observed values (blue) for each of the GP models. To give sense to the plots we use for the X-axis the first components of the corresponding 256 design points. Note that $\xi_i = (\xi_1, \xi_2, \dots, \xi_D)$. The percentage of points out of the range for the SE model is 4.98% and so, 95.02% of the predictions are within the 95% acceptance interval. These results point us out that the appropriate GP emulator for this simulator, from the initial four selected, is the one built up on a mean-zero function and the SE covariance function. Throughout this section we will use this GP emulator and will try to refine it by optimising the number of KL coefficients retained and the number of design points used in the training set.

To select the optimal number of KL coefficients to be retained in each design points, we compute the $NLML(2.14)$, $MSE(2.15)$ and $DS(2.16)$ for the chosen GP model. Figure 2.10 shows the different scores plotted against the number of KL coefficients retained. In this case, all the scores follow the same decreasing tendency when increasing the number of KL coefficients used. The plots show that after around 18 KL coefficients, the scores are not changing significantly. Therefore, it seems sensible to build our training set based on 18-dimensional design points, i.e., $D = 18$ in the emulation diagram 2.5.

Another intuitive and qualitative way of measuring how close our GP predictions are to the observed values is to plot the pairs (*predicted* τ , *observed* τ) along with the line $y = x$ and check that the points are not far away from the straight line (see e.g., Figure 2.11). These plots are also called *scatterplots*. In

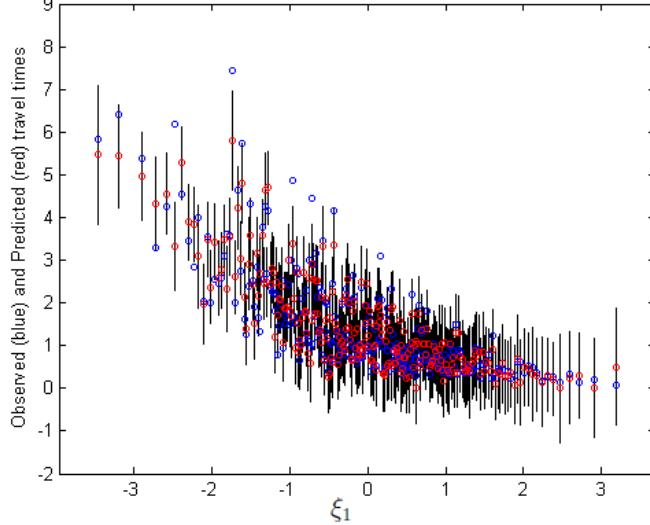


Figure 2.6: Predicted values (red) and their 95% bounds (black bars) given by the emulator against observed values (blue) for the LOO-CV from a design of 256 points and using a mean-zero function and a SE covariance function. 4.98% of the observed values out of range.

this case, given the set of design points, $\{\boldsymbol{\xi}_i, i = 1, \dots, 256\}$, we use as observed the travel time values, $\tau_i = f(\mathbf{K}_i)$, their corresponding predicted values for the same input, $\tau_i^* = \hat{f}(\boldsymbol{\xi}_i)$. The fact that none of the points are away from the straight line could be interpreted as a good sign before doing a proper analysis of the data.

This results corresponds to a design of 256 points but it is reasonable to ask what happens if we use fewer points in our design. For answer that we have computed the relative error¹ between a sample of 1000 observed travel times and the corresponding predictions by using our GP emulator with 1, 2, ..., 256 design points. Figure 2.12 suggests that the adequate number of design points for this problem could be around 64 (power of 2), as taking more points would not improve significantly the GP emulator performance (measured in terms of the RE). The election of the number of design points to keep will depend on

¹The error considered for comparisons between two vectors throughout this thesis will be the L^2 -norm relative error unless stated otherwise. For two vector $\mathbf{x} = (x_1, \dots, x_n)$ and $\mathbf{y} = (y_1, \dots, y_n)$, we define the L^2 -norm relative error between \mathbf{x} and \mathbf{y} as: $RE(\mathbf{x}, \mathbf{y}) = \frac{\|\mathbf{x} - \mathbf{y}\|_2}{\|\mathbf{x}\|_2}$, where $\|\mathbf{x}\|_2$ is Euclidean norm.

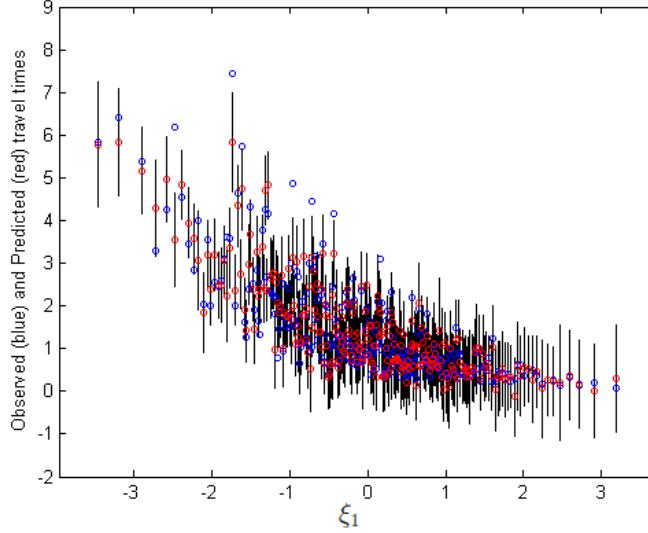


Figure 2.7: Predicted values (red) and their 95% bounds (black bars) given by the emulator against observed values (blue) for the LOO-CV from a design of 256 points and using a mean-zero function and a $\text{Matérn}_{\frac{3}{2}}$ covariance function. 6.64% of the observed values out of range.

the accuracy desired for each problem, for instance, with this (computationally cheap) simulator we can afford to run the model 256 times and therefore keep the whole training set. In others (computationally expensive) simulators, like the one that will be discussed in Chapter 4, a single run could last even a day, and therefore, this consideration about the number of design points to retain becomes extremely important.

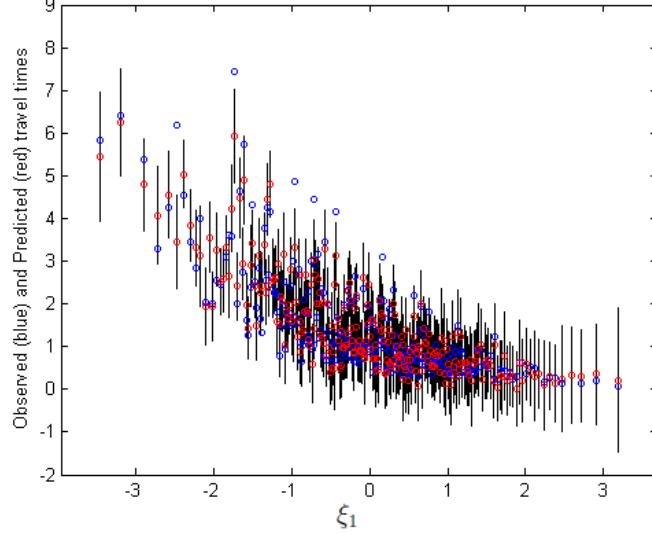


Figure 2.8: Predicted values (red) and their 95% bounds (black bars) given by the emulator against observed values (blue) for the LOO-CV from a design of 256 points and using a mean-zero function and a $\text{Matérn}_{\frac{5}{2}}$ covariance function. 6.25% of the observed values out of range.

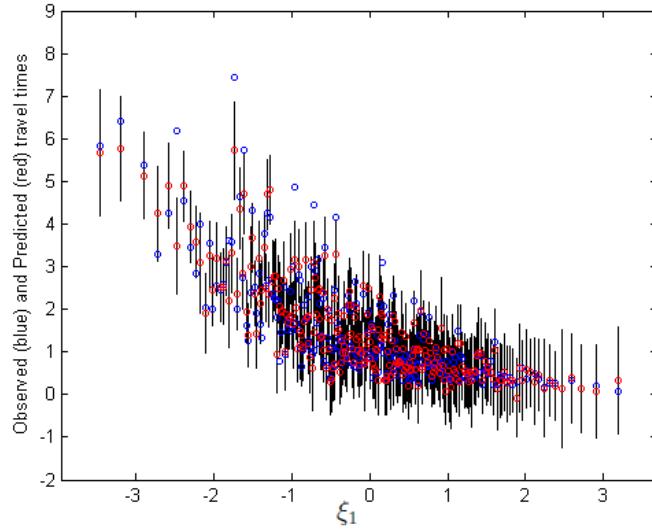


Figure 2.9: Predicted values (red) and their 95% bounds (black bars) given by the emulator against observed values (blue) for the LOO-CV from a design of 256 points and using a mean-zero function and a RQ covariance function. 6.25% of the observed values out of range.

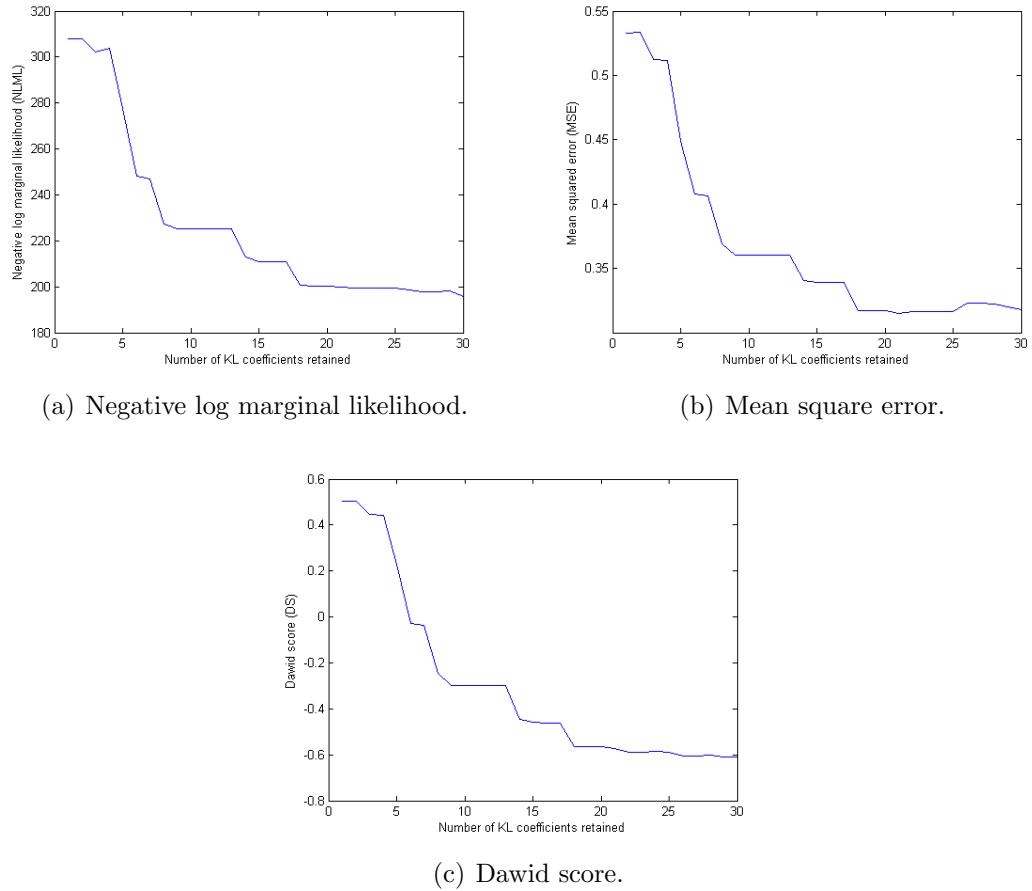


Figure 2.10: Different scores for a GP emulator built with a mean-zero function, SE covariance function and 256 design points. The x-axis represents the number of KL coefficients retained for each score.

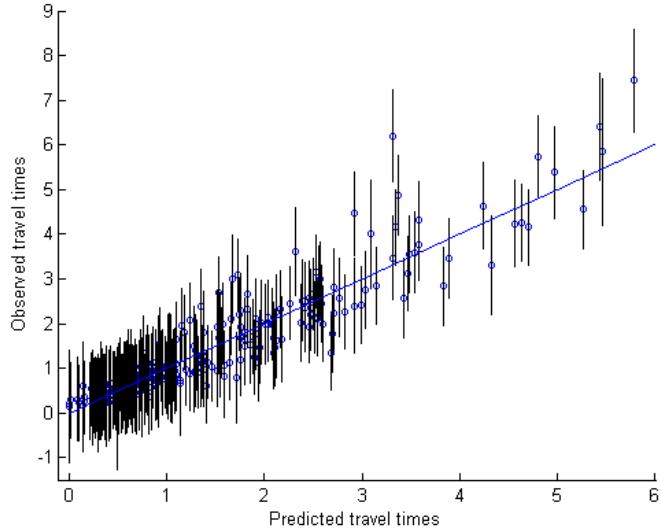


Figure 2.11: Observed and predicted travel times for a design of 256 points, mean-zero function, and $\text{Matérn}_{\frac{3}{2}}$ covariance function. The solid line shows *observed* τ = *predicted* τ .

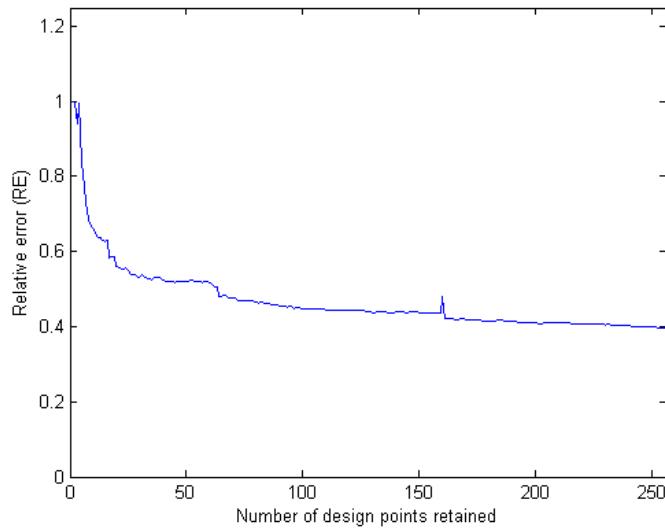


Figure 2.12: Relative error curve between observed and predicted surface fluxes for 256 different designs using *SE* covariance function. The curve shows a smooth decreasing tendency although after around 64 design points retained there is no much improvement in the relative error meanwhile it becomes more computationally expensive.

2.10 Uncertainty quantification of the travel time of a convected particle in groundwater flow

In previous sections we described the mathematical model and numerical code, what we called the simulator, used to simulate the travel time of a convected particle in groundwater flow. In this section, such a simulator will be regarded as a function $f(\cdot)$, that takes a vector \mathbf{x} of *inputs* and produces an *output* scalar $y = f(\mathbf{x})$. We will treat our simulator as deterministic, i.e., it produces the same outputs every time if it is given the same inputs.

The outputs y of a simulator are a prediction of the real-world phenomena that are simulated by the model, but as such will inevitably be imperfect. There will be uncertainty about how close the true real-world quantities that are being approximated by the simulator will be to the outputs y . This uncertainty arises from many sources, particularly uncertainty in the correct values to give the inputs \mathbf{x} and uncertainty about the correctness of the model $f(\cdot)$ in representing the real-world system (O'Hagan, 2004).

Sometimes the values of the inputs are unknown, typically because it is impracticable to measure them. We will consider X , the input configuration to be a random variable with distribution $G(\mathbf{x})$. Consequently the output $Y = f(X)$ is a random variable, and it is the distribution of Y , known as the uncertainty distribution (Oakley and O'Hagan, 2002), that will be of interest. In particular, we are interested in finding the uncertainty distribution of the travel time of a convected particle in groundwater flow induced by the uncertainty distribution of the permeability fields.

To find this distribution, sometimes the complexity of the simulator necessitates the use of approximation methods such as MC as we cannot do the integration analytically. However, the simulator computational complexity limits the number of simulations that can be run, and GP emulation is introduced to help to avoid this computational expense. In the following we will compute the uncertainty distribution of the travel time with the MC method and GP emulation and will compare the results.

2.10.1 Using Monte Carlo method to estimate the cumulative distribution function of the travel time

Suppose we are interested in calculating the cumulative distribution function (CDF) of the quantity of interest, τ , described in Section 2.8.3. We use the MC method described in Section 1.2.2 to approximate that CDF. To do that, we take $T = \tau$ and $\mathbf{X}_M = \mathbf{K}$, then $T_M^{(i)} = \tau_i$. We can now approximate the CDF by the empirical cumulative distribution function (ECDF) of a large sample of τ_i . To compute the ECDF of τ_i we follow these steps:

1. For $i = 1, \dots, n$, simulate a discrete random permeability field \mathbf{K}_i .
2. Use the simulator to compute the corresponding τ_i for each \mathbf{K}_i .

Finally, compute the empirical cumulative distribution function, \hat{F} , of the set of values $\{\tau_i, i = 1, \dots, n\}$,

$$\hat{F}(s) = \frac{1}{n} \sum_{i=1}^n \mathbb{I}_{\{\tau_i \leq s\}},$$

where \mathbb{I} is the *indicator* function,

$$\mathbb{I}_{\{\tau_i \leq s\}} = \begin{cases} 1 & \text{if } \tau_i \leq s, \\ 0 & \text{if } \tau_i > s. \end{cases}$$

Figure 2.13 shows the MC uncertainty analysis for a sample of $n = 50000$ random permeability fields. The blue line is the estimation of the CDF of τ and the dashed lines the 95% uncertainty bounds for this empirical distribution. Unfortunately, for complex simulators, the Monte Carlo approach is too computationally expensive to be applied as the accuracy of \hat{F} depends on the sample size¹ n .

In the following section we will use the GP emulator to perform a full UA of the distribution of the travel time.

¹In this case, the simulator described is computationally cheap enough that sufficient computation can be done within a day. However, this simulator is relatively simple and allows us to develop methodology that could be used for more complex simulators, like the one discussed next in Chapter 4.

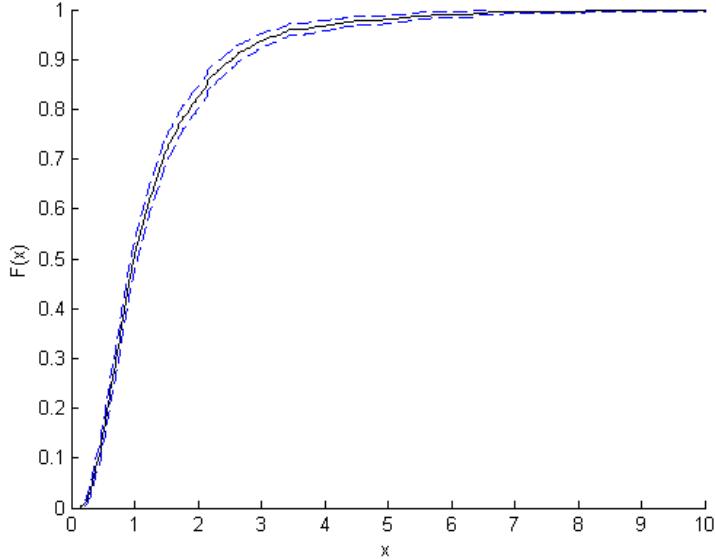


Figure 2.13: Monte Carlo ECDF (black line) based on 50000 travel times. The dashed lines show the 95% uncertainty bounds.

2.10.2 Using Gaussian process emulation to predict the cumulative distribution function of the travel time

To conduct an UA of the travel time with our GP emulator we will follow the methodology proposed by Oakley and O'Hagan (2002). Suppose there is some particular value s of the output (τ) which is considered critical in some sense. It is natural to ask what the probability is that the true output, $Y = f(\mathbf{x})$, will exceed s . This motivates us to consider inference about the distribution function of Y , which we write as

$$F(s) = p(Y \leq s) = \int_{\Omega} \mathbb{I}\{f(\mathbf{x}) \leq s\} dG(\mathbf{x}), \quad (2.36)$$

where Ω is the sample space of any input \mathbf{x} and \mathbb{I} denotes the indicator function. For computational reasons instead of (2.36) we will use the simulation approach described in Oakley and O'Hagan (2002) to derive the posterior moments of $F(\cdot)$. We need first to simulate draws $F_{(i)}(\cdot)$ from the posterior distribution of $F(\cdot)$; To

do this we need to obtain first a realisation of $f_{(i)}(\cdot)$. We do that by drawing a large random sample of inputs $\mathbf{x}_1^*, \dots, \mathbf{x}_M^*$ from $G(\cdot)$, for some integer M . Finally, we take as an approximation for the realisation $f_{(i)}(\cdot)$ its posterior mean, $m_{(i)}^*(\cdot)$, and finally, we approximate $F_{(i)}(\cdot)$ by using the empirical cumulative distribution function

$$F_{(i)}(s) = \frac{1}{M} \sum_{j=1}^M \mathbb{I}\{m_{(i)}^*(\mathbf{x}_j^*) \leq s\}. \quad (2.37)$$

We obtain L_F realisations $F_{(1)}(\cdot), \dots, F_{(L_F)}(\cdot)$, and from this sample of random distribution functions we can obtain any required inference about $F(\cdot)$, for instance the sample mean¹, which is an alternative to (2.36) and it is given by

$$\bar{F}(s) = \frac{1}{L_F} \sum_{j=1}^{L_F} F_{(j)}(s). \quad (2.38)$$

Now to find our uncertainty bounds we consider the corresponding quantile function. If we call p_{α_s} the $100\alpha_s$ percentile, such that $F(p_{\alpha_s}) = \alpha_s$, the distribution of p_{α_s} is given by

$$p(p_{\alpha_s} \leq t) = p\{F(t) \geq \alpha_s\},$$

where $p\{F(t) \geq \alpha_s\}$ can be estimated using the method just described. And then, we can estimate p_{α_s} by its sample mean by finding $p_{(i)\alpha_s}$, the $100\alpha_s$ percentile for realisation i , for $i = 1, \dots, L_F$.

Next we will apply the procedure just described to quantify the uncertainty distribution of the travel time, τ , by using our GP emulator and we will show some of the results obtained.

The procedure followed to perform the GP UA of τ is described next:

1. Generate a large sample of test points $\mathbf{x}_i^* \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, $i = 1, \dots, M$. (M large).
2. Evaluate the emulator (eq. (1.18)) at those \mathbf{x}_i^* , i.e. $f_i = \hat{f}(\mathbf{x}_i^*)$.

¹Oakley and O'Hagan (2002) remarks that since $F(s)$ is constrained to take values between zero and one, the distribution of $F(s)$ may be skewed for low and high values of s . Hence the mean of this distribution may be a poor location summary; it may overestimate $F(s)$ at low values of s and underestimate $F(s)$ at high values of s . Consequently, the sample median might be preferred as a location summary.

3. Store the quantiles for this f_i and form its corresponding ECDF, F_i .

Repeat the process a large number of times, j , and form a family of j distribution functions, F_i^j . Then Compute the lower, upper and mean or median quantiles from this distribution of ECDFs as an approximation of the true CDF of the travel time, τ .

Results of the GP emulation UA are showed in Figures 2.14 and 2.15. Figure 2.14 shows the ECDF (black) based on 50,000 samples of the travel time computed with the MC method and GP posterior samples (green) which cover the travel time ECDF. Figure 2.15 shows the 2.5th and 97.5th percentiles (dashed magenta), the GP posterior mean (red) of the cumulative distribution function, and MC ECDF (black line).

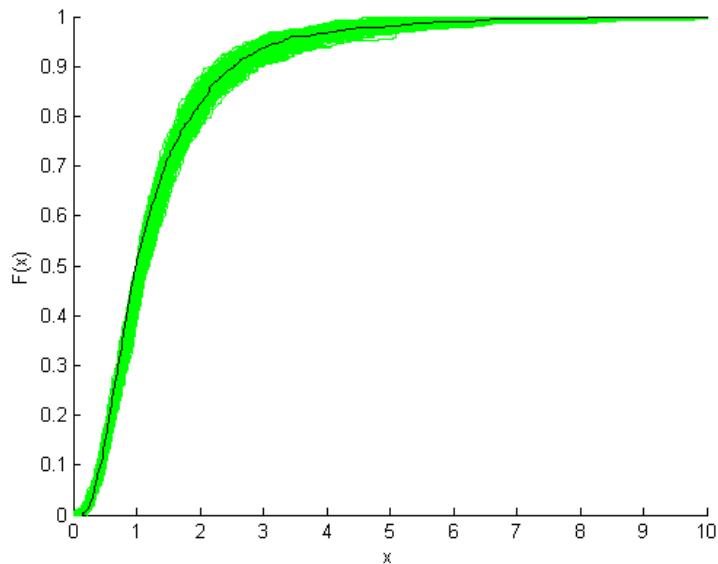


Figure 2.14: GP Posterior ECDFs samples, F_i , approximating the MC estimation of the CDF of the travel time of a convected particle in groundwater flow. The GP emulator prior specifications are: mean-zero function, SE covariance function, 256 design points and 16 KL coefficients retained.

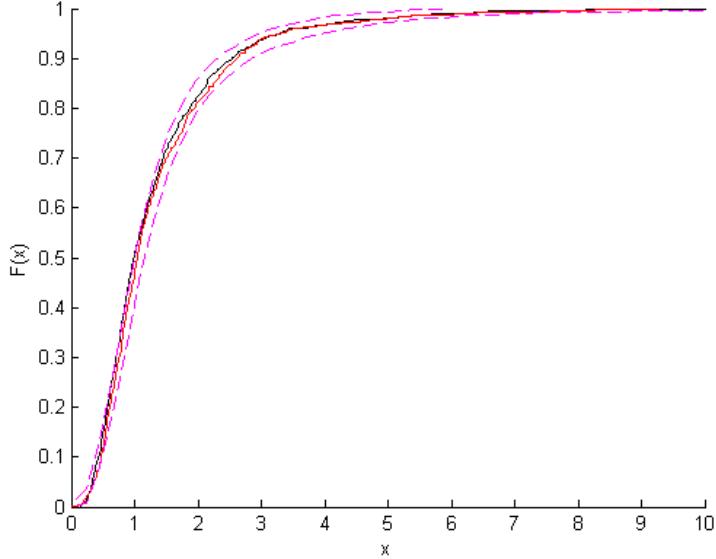


Figure 2.15: GP uncertainty analysis of the CDF of the travel time of a convected particle in groundwater flow. The GP emulator prior specifications are: mean-zero function, SE covariance function, 256 design points and 16 KL coefficients retained.

So far, we have been using our GP emulator as a tool for predicting scalar values for different collections of discrete random permeability fields, in other words, we have built the GP emulator as a map from \mathbb{R}^d to \mathbb{R} . In the last part of this chapter we present a different technique for estimating the travel time τ . In this technique we build a GP to emulate the pressure field first, and then we obtain τ from that emulated field by using equation (2.23). We will show the comparison between this method and the standard scalar emulation presented previously in order to see which method is more efficient.

2.11 Gaussian process emulation of two-dimensional fields

Suppose we have a simulator that takes input \mathbf{x} , i.e. the coefficients $\{\xi_1, \dots, \xi_M\}$, of the KL decomposition (2.32), and returns an output \mathbf{y} , where \mathbf{y} is a 2-dimensional field, for instance, a pressure field solution of (2.35). The goal is to build an emulator from \mathbf{x} to \mathbf{y} , but the high dimension of \mathbf{y} means that standard approaches, as used in Section 1.3.2, will not work. So, the idea is to obtain a reduced rank approximation, using a matrix decomposition of the output. One way to do this is with principal component analysis (PCA) (Wilkinson (2011), chapter 10), or with the singular value decomposition (SVD) (Holden et al., 2015). In this thesis we will use the later approach.

2.11.1 The singular value decomposition method

Let us describe how we apply the SVD method to our simulator-emulator:

1. Let $\mathbf{y}_1, \dots, \mathbf{y}_n$ be n pressure fields obtained from running the simulator n times. Assume each of these is a vector of length M (if the pressure field is a matrix, stack the columns to form a vector).
2. Let Y be the $M \times n$ matrix with column j being \mathbf{y}_j .
3. Let $\tilde{Y} = \tilde{\mathbf{y}}_1, \dots, \tilde{\mathbf{y}}_n$ be the row centered¹ version of Y .
4. Compute the SVD of \tilde{Y}

$$\tilde{Y} = LDR^\top$$

where L is a $M \times n$ matrix containing the left singular vectors, D is a $n \times n$ matrix containing the singular values, and R is an $n \times n$ matrix containing the right singular vectors.

¹The row centered version of a matrix Y is obtained by subtracting the mean of row i from every element in row i , so that the mean of \tilde{Y}_i is equal to zero.

Note that $L = (\mathbf{l}_1, \dots, \mathbf{l}_n)$, where $\mathbf{l}_j \in \mathbb{R}^M$ is the j^{th} eigenvector of $\tilde{Y}\tilde{Y}^\top$, $D_* = (\mathbf{d}_1, \dots, \mathbf{d}_n)$, where $\mathbf{d}_j \in \mathbb{R}^n$, and $R = (\mathbf{r}_1, \dots, \mathbf{r}_n)$, where $\mathbf{r}_j \in \mathbb{R}^n$ is the j^{th} eigenvector of $\tilde{Y}^\top\tilde{Y}$.

5. We can form a reduced rank approximation to \tilde{Y} by ignoring all but the first k ($k < n$) eigenvectors, i.e, let:

$$L_* = (\mathbf{l}_1, \dots, \mathbf{l}_n), \quad D_* = (\mathbf{d}_1, \dots, \mathbf{d}_k), \quad R_* = (\mathbf{r}_1, \dots, \mathbf{r}_k)$$

so that

$$\tilde{Y} \approx L_* D_* R_*^\top.$$

If we consider $R_*^\top = (\mathbf{t}_1, \dots, \mathbf{t}_n)$, where each \mathbf{t}_j is a vector of length k , then we can approximate the j^{th} centered field of \tilde{Y} , $\tilde{\mathbf{y}}_j$, with

$$\tilde{\mathbf{y}}_j \approx \tilde{\mathbf{y}}_j^* = L_* D_* \mathbf{t}_j. \quad (2.39)$$

To quantify the accuracy of the reduced rank approximation according to the number of eigenvectors k retained, we can use the RE between the initial centered field $\tilde{\mathbf{y}}_j$ and its reduced rank approximation $\tilde{\mathbf{y}}_j^*$, for any given $j \in \mathbb{N}$. Suppose we want to find number of eigenvectors k to retain which guarantees¹ an overall RE for all the reduced rank approximations, i.e, for a fixed desired tolerance, TOL , we can compute the number of k needed to assure,

$$RE(\tilde{\mathbf{y}}_1, \tilde{\mathbf{y}}_1^*) < TOL, \quad RE(\tilde{\mathbf{y}}_2, \tilde{\mathbf{y}}_2^*) < TOL, \quad , \dots, \quad RE(\tilde{\mathbf{y}}_n, \tilde{\mathbf{y}}_n^*) < TOL,$$

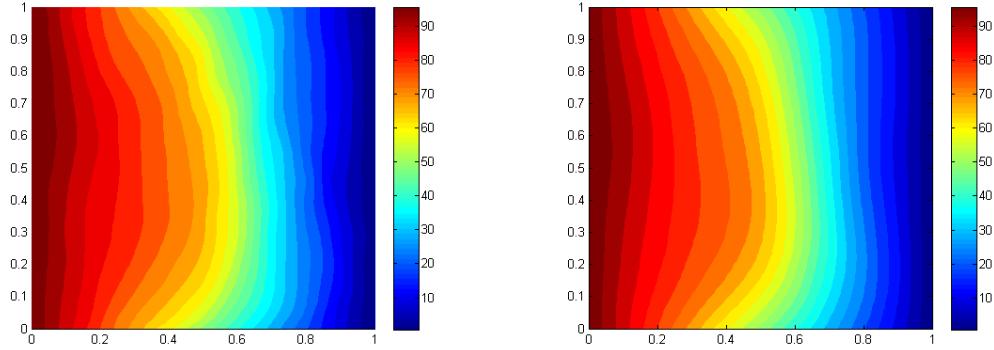
and so, keep this k to form the desired reduced rank approximation with the SVD method. Let us see an example.

2.11.2 Reduced rank emulation of pressure fields

For this example we will consider the set of 256 pressure fields, $\{P_1, P_2, \dots, P_{256}\}$, obtained by running the simulator at the 256 design points used in Section 2.9.1,

¹Note that this is always possible since we could take $k = n$ which guarantees a $RE = 0$.

i.e., we let $\mathbf{y}_1 = P_1$, $\mathbf{y}_2 = P_2, \dots$, $\mathbf{y}_n = P_n$, and $n = 256$. We apply the method just described in Section 2.11.1 and compute two approximations. Figures 2.16(a) and 2.17(a) show the true pressure fields, P_1 and P_2 , and Figures 2.16(b) and 2.17(b) the corresponding reduced rank approximations, \tilde{P}_1 and \tilde{P}_2 . The RE for P_1 and \tilde{P}_1 is 0.00042 by retaining $k = 12$ eigenvectors and 0.00581 for P_2 and \tilde{P}_2 by retaining $k = 16$.

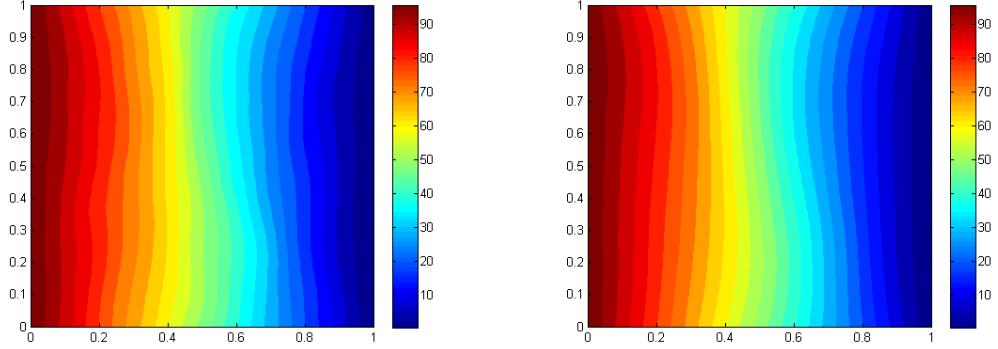


(a) True pressure field, P_1 , obtained from the simulator. (b) Reduced rank approximation, \tilde{P}_1 , of the pressure field P_1 by using 16 eigenvectors.

Figure 2.16: True pressure field, P_1 , and corresponding reduced rank approximation, \tilde{P}_1 . Number of eigenvectors used is $k = 12$. The RE between P_1 and \tilde{P}_1 is 0.00042.

Figure 2.18 shows the RE between all the 256 pressure fields and their corresponding reduced rank approximations (Y-axis) and the number of eigenvectors retained (X-axis). Note that after 130 eigenvectors retained the RE is almost zero.

We will show in the following section how we combine the SVD method with GP emulation to make pressure field predictions.



(a) True pressure field, P_2 , obtained from the simulator.
(b) Reduced rank approximation, \tilde{P}_2 , of the pressure field P_2 by using 16 eigenvectors.

Figure 2.17: True pressure field, P_2 , and corresponding reduced rank approximation, \tilde{P}_2 . Number of eigenvectors used is $k = 16$. The RE between P_2 and \tilde{P}_2 is 0.00581.

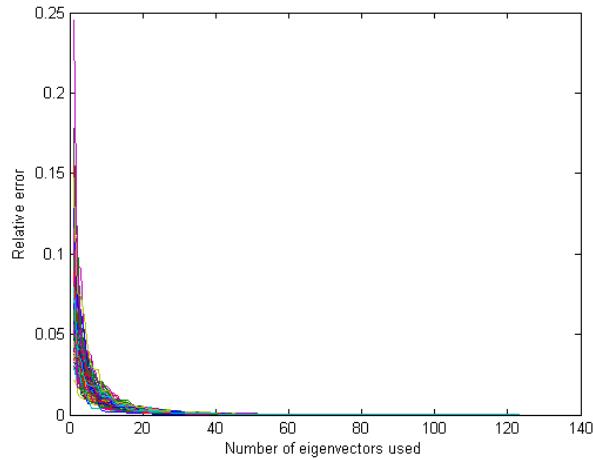


Figure 2.18: RE (Y axis) between each of the true pressure fields and their corresponding reduced rank approximations along the number of eigenvectors retained (X axis).

2.11.3 Building the Gaussian process field emulator

Let $f(\cdot)$ be now the pressure field simulator¹, i.e., the simulator which takes an input \mathbf{x} , and returns an output \mathbf{y} , where \mathbf{y} is a 2-dimensional pressure field solution of (2.35). Suppose we have a reduced rank approximation of n fields, $\mathbf{y}_1, \dots, \mathbf{y}_n$, given by expression

$$\tilde{\mathbf{y}}_j \approx L_* D_* \mathbf{t}_j, \quad \text{for } i = 1, \dots, n, \quad (2.40)$$

where $L_* = (\mathbf{l}_1, \dots, \mathbf{l}_n)$ and $D_* = (\mathbf{d}_1, \dots, \mathbf{d}_k)$, for some $k < n$, and \mathbf{t}_j is a vector of length k (as discussed in Section 2.11.1). To build an emulator, f^* , for the simulator, we can build an emulator by using \mathbf{x} and the n vectors \mathbf{t}_j as a training set. To do so, we build k separate emulators, f_j^* , for $j = 1, \dots, k$, with their respective training sets formed by \mathbf{x} and each of the elements of the rows of the matrix $R_*^\top = (\mathbf{t}_1, \dots, \mathbf{t}_n)$. For a new given input \mathbf{x}^* we can compute $f_j^*(\mathbf{x}^*) := t_j^*$, for $j = 1, \dots, k$, and form $f^*(\mathbf{x}^*) := (f_1^*(\mathbf{x}^*), \dots, f_k^*(\mathbf{x}^*)) = \mathbf{t}^*$, where $\mathbf{t}^* := (t_1^*, \dots, t_k^*)$. So, given \mathbf{t}^* , the estimated centered field will be given by,

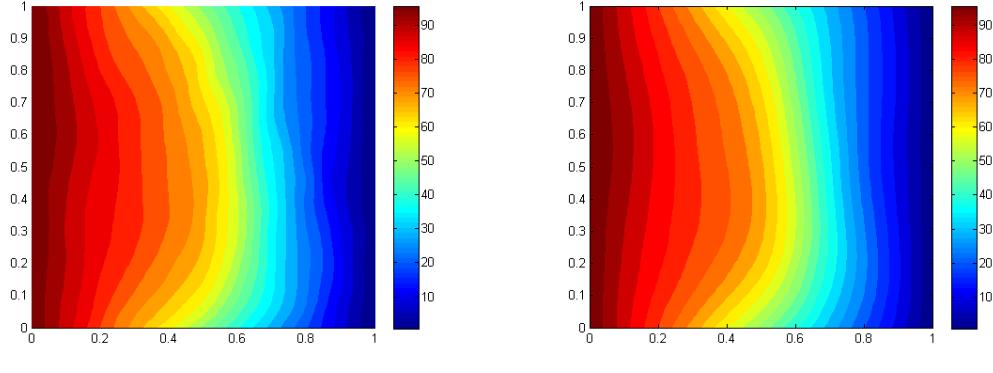
$$\tilde{y}^* = f^*(\mathbf{x}^*) = L_* D_* \mathbf{x}^*. \quad (2.41)$$

Figure 2.19 shows one example of a predicted pressure field with the GP field emulator. We can see the differences² among the observed pressure field, P , the corresponding reduced rank approximation, \tilde{P} , and the predicted pressure field, P^* . The number of eigenvectors used for this example is $k = 40$ and the number of KL coefficients retained is 16. The L^2 -norm relative error between P and \tilde{P} is 0.00042 and the L^2 -norm relative error between P and P^* is 0.01594.

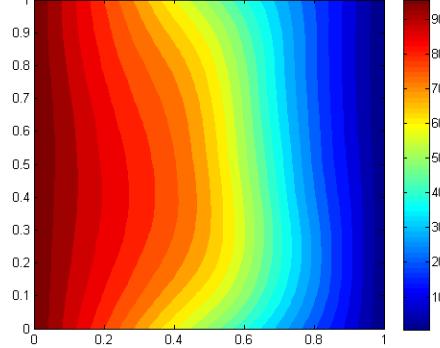
We will finish this chapter with a comparison between the two approaches

¹So far, we have been using the term *simulator* to the numerical code which takes an input (permeability field) and returns a scalar output, the travel time. We will call *field simulator* the numerical code which takes as input a permeability field and returns as output the pressure field solution of equation (2.35).

²Note that although the relative errors are measured always between the true and the predicted field, the prediction cannot be ever better than the reduced rank approximation since the GP emulator is built by using the training set given by actually the reduced rank approximation.



(a) Observed pressure field, P , obtained from the simulator for the elliptic problem. (b) Reduced rank approximation, \tilde{P} , of the observed pressure field P by using 40 eigenvectors.



(c) Predicted pressure field, P^* , of the concentration field P by using 40 eigenvectors and 16 KL coefficients.

Figure 2.19: Observed pressure field, P , and corresponding reduced rank approximation, \tilde{P} , and predicted pressure field, P^* . Number of eigenvectors used is $k = 40$. Number of KL coefficients for prediction is 16. The L^2 -norm relative error between P and \tilde{P} is 0.00042 and the L^2 -norm relative error between P and P^* is 0.01594.

described for making predictions on the travel time of a convected particle in groundwater flow, namely, the GP scalar emulator and the GP field emulator.

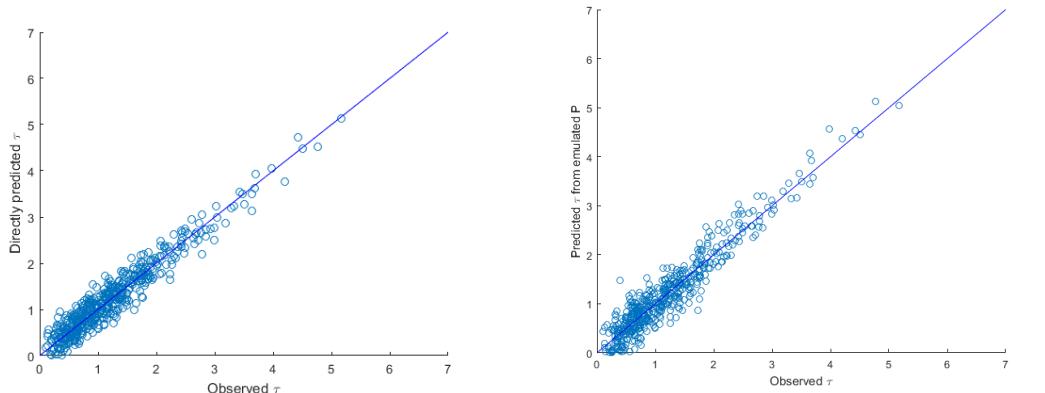
2.12 Comparison between direct travel time emulation and travel time obtained from pressure field emulation

The GP prior specifications for all the emulators will be the same used throughout this chapter, i.e., zero-mean function, SE covariance function, a training set formed by 256 points, and 16 KL coefficients retained. The number of eigenvectors used for this comparison is $k = 40$.

For this comparison we generate 500 permeability fields, \mathbf{K}_i , and use the simulator to compute the corresponding 500, τ_i , outputs from problem (2.35) which will be considered the true values. Then, we use the GP scalar emulator and the GP field emulator to predict the the corresponding 500 travel times. With these specifications the RE for the scalar GP when compared with the simulator was 0.0182 meanwhile the RE for the field GP emulator was 0.0201.

The conclusion for this experiment is that scalar GP emulation seems to be more efficient than field GP emulation, although both of them seem to be a powerful tool for making predictions for problem (2.35). Figures 2.20(a) and 2.20(b) show visual comparison of both results. One of the possible reasons why those differences between the results occur, might be that field emulation is loosing accuracy along with the increment of the degrees of freedom (i.e., the number of eigenvector retained) in the SVD process. Other possibility, although it has not measure properly in this thesis, might be that the relative error in field emulation increases along with the number of scalar emulations required (one per each degree of freedom).

From this above we can conclude that scalar emulation is recommended when looking for accuracy in predicting scalar outputs, whereas field emulation is clearly needed when attempting to predict the physical process for a given problem.



(a) Comparison of 500 Observed and directly predicted τ . The RE between the observed and predicted was 0.0182.

(b) Comparison of 500 Observed τ and τ from emulated pressure field. The number of eigenvectors used during the reduced rank approximation was 40. The RE between the observed and predicted was 0.0201.

Figure 2.20: Observed τ vs directly predicted SF and Observed τ vs τ from predicted pressure field plots obtained from the simulator. The design was formed by 256 points. A zero-mean and SE covariance function were used as prior GP model specifications and the number of KL coefficients used for prediction was 16. The solid line shows $observed \tau = predicted \tau$.

Chapter 3

Comparison of GP emulation with other methods for UQ in groundwater flow problems

In this chapter we will use the methodologies introduced in Section 1.2.1, namely, MC, MLMC, QMC and MLQMC methods, and GP emulation, discussed in Chapter 2, to estimate the average travel time of a particle convected in groundwater flow. We will show some comparisons of the estimates obtained with the above methods based on different discretisations of the physical domain. The performance of the MC methods will be tested by comparing their computational ε^2 -cost, i.e., the number of floating point operations that are needed to achieve the desired MSE. The results obtained with the GP emulator will be given in terms of 95% confidence intervals (CI).

3.1 Comparison among Monte Carlo methods for solving PDEs with random inputs

The aim of this study is to analyse alternatives to the standard MC method for solving PDEs with random inputs. We will use all the methods described in Section 1.2.1 to compute the average travel time of a particle convected in groundwater flow. For this purpose, we use the travel time simulator (2.8.3) for the model problem (2.35) in $D = [0, 1] \times [0, 1] \subset \mathbb{R}^2$, with $h \equiv 0$, and boundary conditions,

$$\begin{aligned} p(0, y) &= 250, & p(1, y) &= 0, \\ \frac{\partial p}{\partial y}(x, 0) &= 0, \\ \frac{\partial p}{\partial y}(x, 1) &= 0. \end{aligned}$$

The permeability fields are generated as discussed in Section 2.8.5 with parameters, $\lambda = 0.3$ and $\sigma = 1.0$ (as suggested in (Cliffe et al., 2011)).

The sequence of levels (see Section 1.2.3) will start with $M_0 = 64$, this enables to get a minimal level of resolution of the problem (Giles (2008) and Cliffe et al. (2011)). The maximum level considered will be $M_5 = 65536$ grid points.

Cliffe et al. (2011) states that the particular choice of spatial discretisation scheme is not essential to the MLMC approach. However, many quantities of interest in subsurface flow depend on an accurate and mass-conservative representation of the flux, and so in this context finite volume (FV) or mixed finite elements (FEs) are usually preferred over other methods. In this thesis, due to the simplicity of the computational domain (square and with equally spaced nodes), the travel time simulator will use a standard cell-centred FV method (for details about the discretisation scheme, see (Cliffe et al., 2011)).

The assumptions of Theorem 1.2.1 for the mean and the variance of the MLMC (1.9) and MLQMC (1.16) estimators will be numerically confirmed for each of the cases. The estimates of the parameters α and β will be computed “on the fly” from sample averages. For simplicity, we will assume that $\gamma = 1$ in all the

simulations¹. To quantify the cost of the algorithms, we will assume that the number of operations to compute one sample on level ℓ is $C_\ell = cM_\ell^\gamma$ for some fixed constant c , as discussed in Section 1.2.5, and thus, in the results presented in this thesis, we will show the standardised costs, scaled by $1/c$.

In the following sections we show a series of comparisons between the above methods according to different tolerances measured by the MSE. The average travel times estimated with MC, MLMC, QMC and MLQMC methods will be denoted by T_{MC} , T_{MLMC} , T_{QMC} and T_{MLQMC} respectively.

3.1.1 Comparison between classical MC and MLMC

In this section we compare the performance of MC and MLMC methods for the following MSEs: 0.01, 0.0064 and 0.0025.

Tables 3.1, 3.2, 3.3 show the number of samples, N_ℓ , used by the MLMC method in each level, ℓ , for the given MSE, ε^2 , the value of the average travel time, T_{MLMC} , the corresponding bounds for the estimation, $(T_{MLMC} - \varepsilon, T_{MLMC} + \varepsilon)$, and the final computational ε^2 -Cost spent for each tolerance.

Figure 3.1 shows the expected value of T_ℓ and $Y_\ell = T_\ell - T_{\ell-1}$ and how the slope of the line for $\mathbb{E}[T_\ell - T_{\ell-1}]$ has a decreasing tendency. It also shows how $\mathbb{E}[T_\ell]$ is approximately constant on all levels, numerically verifying the assumption made in Theorem 1.2.1.

Figure 3.2 shows the behaviour of the variance of T_ℓ and $Y_\ell = T_\ell - T_{\ell-1}$ for each level ℓ , and how it is numerically satisfied the assumption of a constant variance on all levels.

¹Note that for this simulator, applied to a 2D problem and implemented in Matlab, a value of $\gamma = 1.5$ could be more appropriate (Cliffe et al., 2011), although it is not relevant for our comparison purpose.

Level ℓ	Number of Samples N_ℓ	ε^2 - Cost ($\varepsilon^2 = 0.01$)	T_{MLMC}	MLMC bounds
0	704			
1	93			
2	19			
3	9	86,784	1.3520	(1.2520, 1.4520)

Table 3.1: MLMC estimation with bounds of the average travel time according to a given $MSE = 0.01$. The last number of the first column shows the level at which the code stops.

Level ℓ	Number of Samples N_ℓ	ε^2 - Cost ($\varepsilon^2 = 0.0064$)	T_{MLMC}	MLMC bounds
0	1,103			
1	147			
2	26			
3	12			
4	6	222,223	1.3615	(1.2815, 1.4415)

Table 3.2: MLMC estimation with bounds of the average travel time according to a given $MSE = 0.0064$. The last number of the first column shows the level at which the code stops.

Level ℓ	Number of Samples N_ℓ	ε^2 - Cost ($\varepsilon^2 = 0.0025$)	T_{MLMC}	MLMC bounds
0	3,458			
1	613			
2	104			
3	27			
4	10			
5	5	908,226	1.3696	(1.3196, 1.4196)

Table 3.3: MLMC estimation with bounds of the average travel time according to a given $MSE = 0.0025$. The last number of the first column shows the level at which the code stops.

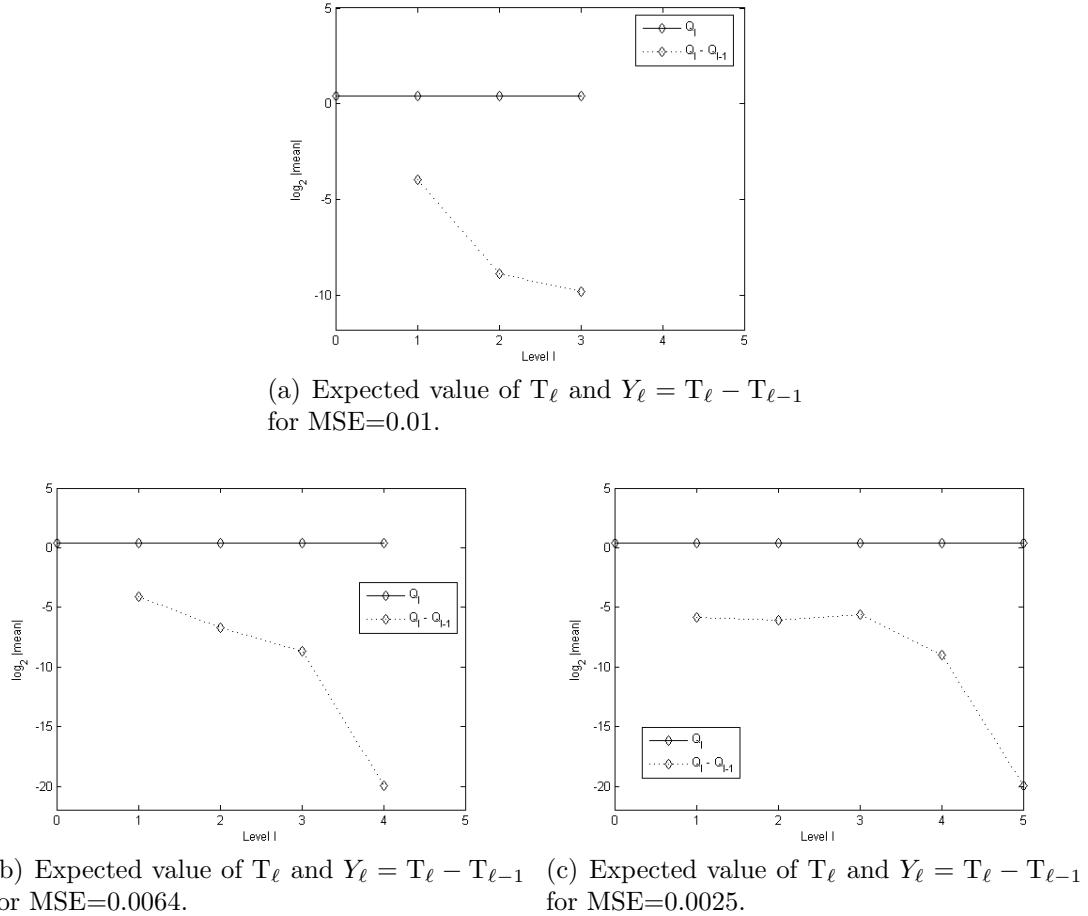


Figure 3.1: Performance plots for the expectation in the MLMC method. The plots show the numerical verification of the asymptotic behaviour of the expectation of T and the convergence of $\mathbb{E}[Y_\ell]$.

3.1.2 Comparison between QMC and MLQMC

In this section we compare the performance of QMC and MLQMC methods for the same MSEs than above. We illustrate next the same tables and figures showed in the previous section for MC and MLMC methods.

Tables 3.4, 3.5, 3.6 show the number of samples, N_ℓ , used by the MLQMC method in each level, ℓ , for the given MSE, ε^2 , the value of the average travel time, T_{MLQMC} , the corresponding bounds for the estimation, $(T_{MLQMC} - \varepsilon, T_{MLQMC} +$

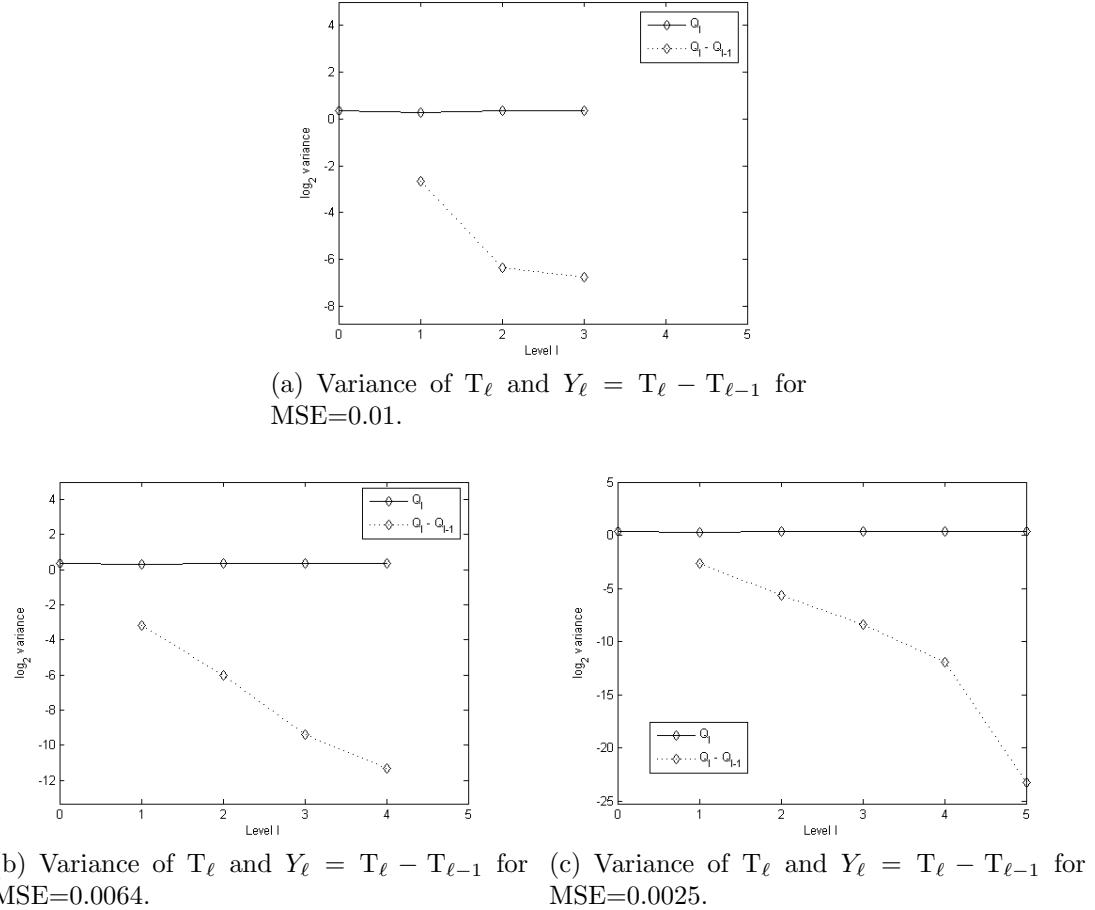


Figure 3.2: Performance plots for the variance in the MLMC method. The plots show the numerical verification of the asymptotic behaviour of the variance of T and the convergence of $\mathbb{V}[Y_\ell]$.

ε), and the final computational ε^2 -Cost spent for each tolerance.

Figure 3.3 shows the expected value of T_ℓ and $Y_\ell = T_\ell - T_{\ell-1}$ and how the slope of the line for $\mathbb{E}[T_\ell - T_{\ell-1}]$ has a decreasing tendency. It also shows how $\mathbb{E}[T_\ell]$ is approximately constant on all levels.

Figure 3.4 shows the behaviour of the variance of T_ℓ and $Y_\ell = T_\ell - T_{\ell-1}$ for each level ℓ , and how it is numerically satisfied the assumption of a constant variance on all levels.

Level ℓ	Number of Samples N_ℓ	ε^2 - Cost ($\varepsilon = 0.01$)	T_{MLQMC}	MLQMC bounds
0	488			
1	60			
2	11			
3	10	71,912	1.2985	(1.1985, 1.3985)

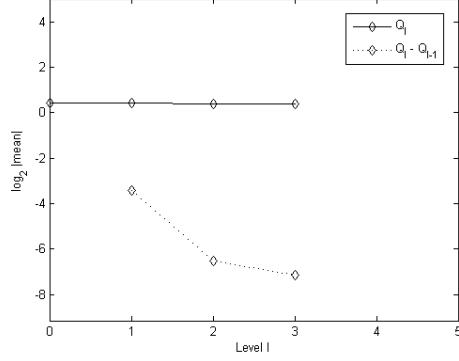
Table 3.4: MLQMC estimation with bounds of the average travel time according to a given $\text{MSE} = 0.01$. The last number of the first column shows the level at which the code stops.

Level ℓ	Number of Samples N_ℓ	ε^2 - Cost ($\varepsilon = 0.005$)	T_{MLQMC}	MLQMC bounds
0	824			
1	109			
2	11			
3	9			
4	4	149,368	1.3427	(1.2627, 1.4227)

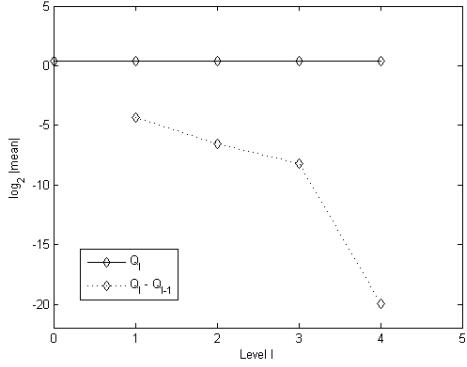
Table 3.5: MLQMC estimation with bounds of the average travel time according to a given $\text{MSE} = 0.0064$. The last number of the first column shows the level at which the code stops.

Level ℓ	Number of Samples N_ℓ	ε^2 - Cost ($\varepsilon = 0.0025$)	T_{MLQMC}	MLQMC bounds
0	2,740			
1	389			
2	57			
3	10			
4	10			
5	5	718,068	1.3550	(1.3005, 1.4050)

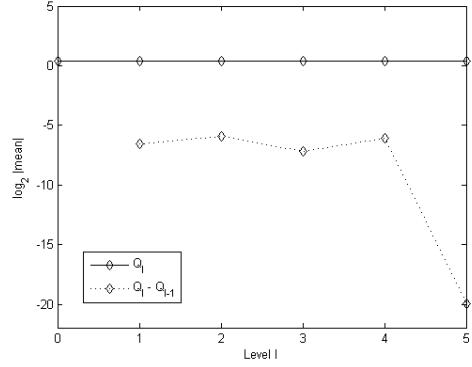
Table 3.6: MLQMC estimation with bounds of the average travel time according to a given $\text{MSE} = 0.0025$. The last number of the first column shows the level at which the code stops.



(a) Expected value of T_ℓ and $Y_\ell = T_\ell - T_{\ell-1}$ for $\text{MSE}=0.01$.



(b) Expected value of T_ℓ and $Y_\ell = T_\ell - T_{\ell-1}$ for $\text{MSE}=0.0064$.



(c) Expected value of T_ℓ and $Y_\ell = T_\ell - T_{\ell-1}$ for $\text{MSE}=0.0025$.

Figure 3.3: Performance plots for the expectation in the MLQMC method. The plots show the numerical verification of the asymptotic behaviour of the expectation of T and the convergence of $\mathbb{E}[Y_\ell]$.

3.1.3 Comparison between MC and QMC

In this section we compare the performance of MC and QMC methods for the same tolerances used in the previous sections.

Table 3.7 shows the comparison of the computational ε^2 -Cost for MC and QMC methods obtained previously in the corresponding MLMC and MLQMC simulations. To calculate the costs for MC and QMC methods we use the esti-

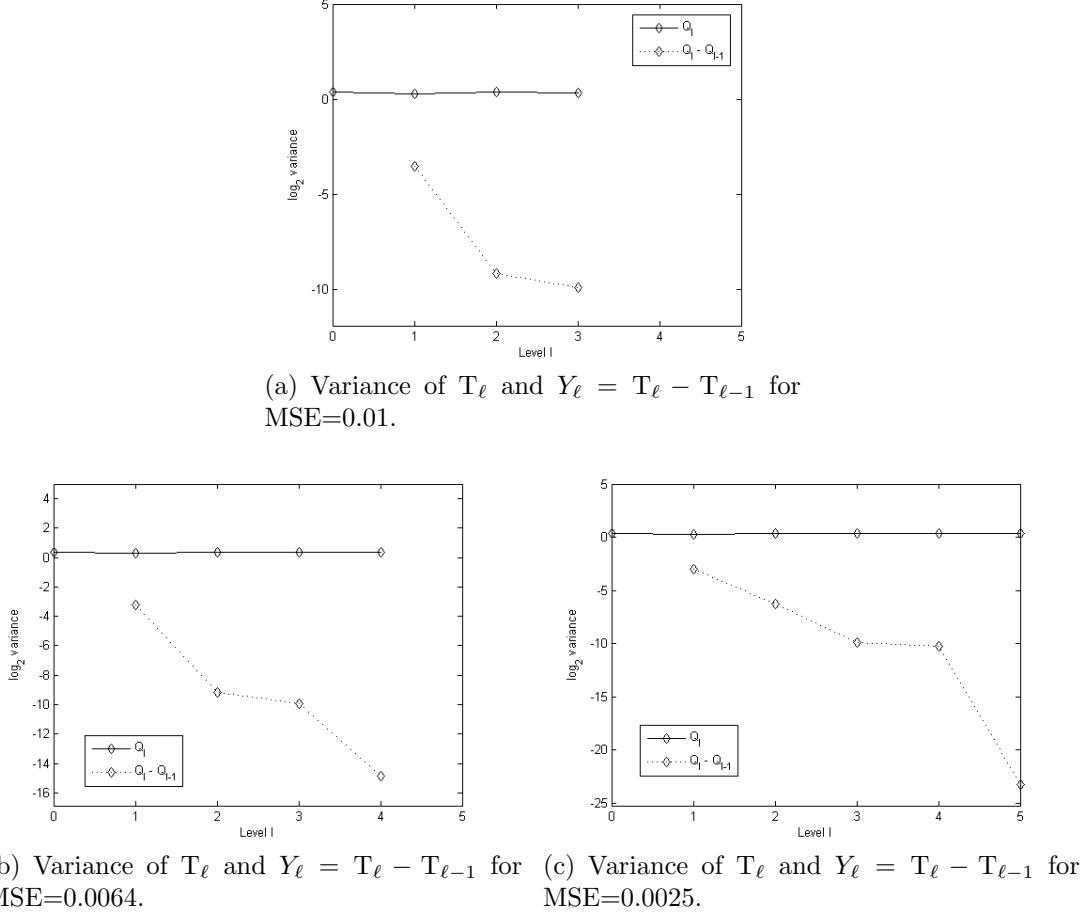


Figure 3.4: Performance plots for the variance in the MLQMC method. The plots show the numerical verification of the asymptotic behaviour of the variance of T and the convergence of $\mathbb{V}[Y_\ell]$.

mator provided in (Giles, 2008), namely,

$$C^* = \sum_{\ell=0}^L N_\ell^* M_\ell, \quad (3.1)$$

where $N_\ell^* = 2\varepsilon^{-2}\mathbb{V}[T_\ell]$, so that the variance of the MC (1.3) and QMC (1.15) estimators is $\frac{1}{2}\varepsilon^2$ as with the corresponding MLMC and MLQMC methods.

In addition to this ε^2 -Cost comparison, we will analyse the convergence of MC and QMC methods at each of the levels where the multilevel methods converged. Figure 3.5 shows the convergence of MC and QMC methods for the average travel

time at levels 3, 4 and 5.

Level ℓ	MSE (ε^2)	ε^2 -Cost MC	ε^2 -Cost QMC
3	0.01	1,746,850	582,980
4	0.0064	18,090,227	1,907,358
5	0.0025	23,650,623	13,299,654

Table 3.7: MC and QMC ε^2 -Cost, obtained from the MLMC and MLQMC simulations respectively, according to the given MSE.

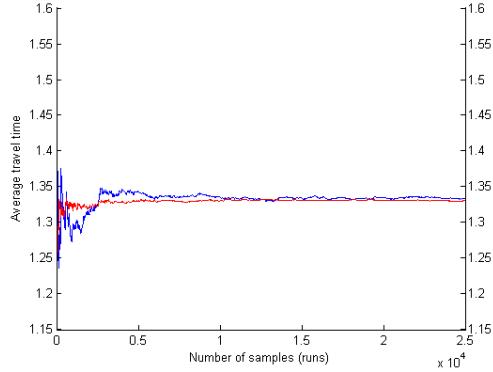
Level ℓ	T_MC samples	T_QMC samples
3	1.3255	1.3305
4	1.3312	1.3299
5	1.3253	1.3262

Table 3.8: Comparison of the travel time estimations obtained with MC and QMC methods at each level based on 25,000 travel time samples.

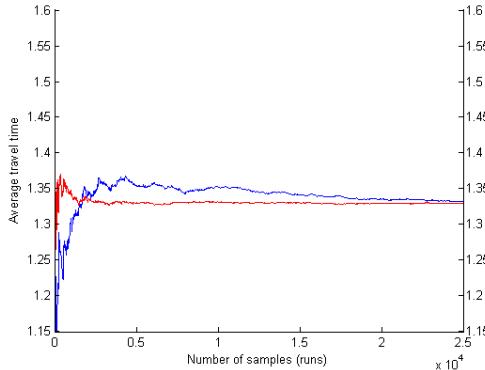
3.1.4 Comparison of MC, QMC, MLMC and MLQMC

The overall picture with the performance of all the methods is showed in Figure 3.6. We can see how MLQMC method produce the lower computational costs for all the tolerances. MLMC is performing better than MC and QMC, and in conclusion, MC seems to be the less efficient method.

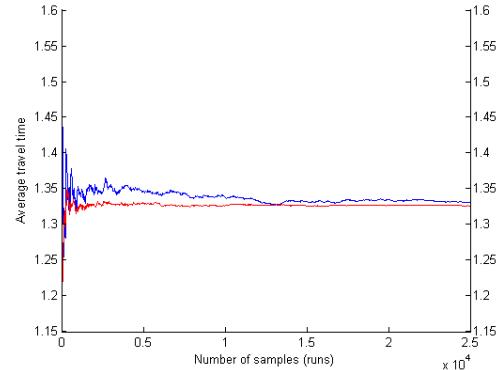
In the last part of this chapter, we will use the GP emulator built in Chapter 2 to predict the average travel time for the same three levels where the other methods converged.



(a) MC and QMC convergence at level 3.



(b) MC and QMC convergence at level 4.



(c) MC and QMC convergence at level 5.

Figure 3.5: Analysis of the convergence of MC and QMC methods for the average travel time at levels 3, 4 and 5. The convergence is calculated over a sample of 25,000 travel times.

3.1.5 Comparison of MC and GP emulation

In this section we will show the predictions for the travel time given by the GP emulator with a zero-mean function and three different covariance functions, namely, SE , $Matérn_{\frac{3}{2}}$ and RQ . The corresponding predictions values with those specifications will be denoted, respectively, by T_{GP}^{SE} , $T_{GP}^{3/2}$ and T_{GP}^{RQ} . We will also show the 95% CI for those values. For all the models we have used 256 design points and retained 18 KL coefficients.

Tables 3.9, 3.10 and 3.11 show the predictions for the average travel time and

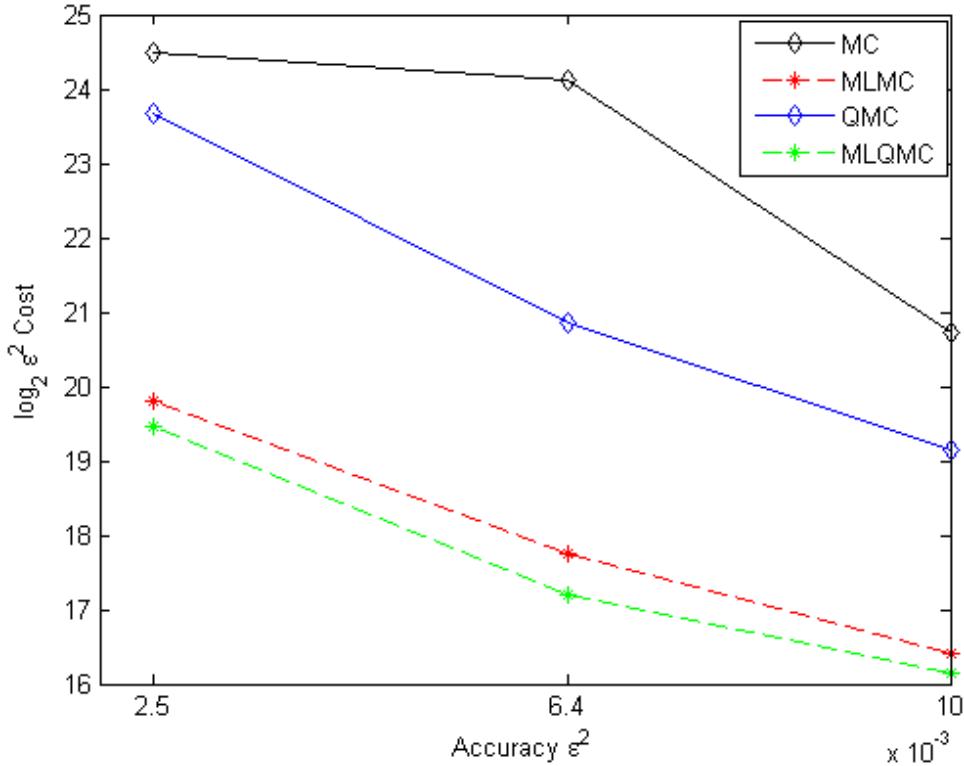


Figure 3.6: ε^2 -Cost for MC, QMC, MLMC and MLQMC methods for MSE: 0.01, 0.0064 and 0.0025.

confidence interval for the three GP models.

To finish this chapter we will show in Figure 3.7 a general picture of the convergence of the MC method for the previous tolerances, and the corresponding estimations of the average travel time for the three GP emulators at each level. We see how all the predictions lie within the RMSE bars.

Level ℓ	T_{GP}^{SE}	95% CI
3	1.3294	(1.3210, 1.3378)
4	1.3336	(1.3247, 1.3425)
5	1.3482	(1.3394, 1.3569)

Table 3.9: Prediction of the average travel time with the GP emulator for levels 2, 3, 4 and 5. A zero-mean and SE covariance functions are used in this model. The training set is formed by 256 points, and the number of KL coefficients retained is 18. The respective 95% confidence intervals for each of the predictions are shown in right column.

Level ℓ	$T_{GP}^{3/2}$	95% CI
3	1.3302	(1.3216, 1.3388)
4	1.3036	(1.2948, 1.3124)
5	1.3301	(1.3215, 1.3388)

Table 3.10: Prediction of the average travel time with the GP emulator for levels 2, 3, 4 and 5. A zero-mean and $Matérn_{\frac{3}{2}}$ covariance functions are used in this model. The training set is formed by 256 points, and the number of KL coefficients retained is 18. The respective 95% confidence intervals for each of the predictions are shown in right column.

Level ℓ	T_{GP}^{RQ}	95% CI
3	1.3174	(1.3091, 1.3258)
4	1.2956	(1.2868, 1.3045)
5	1.3329	(1.3242, 1.3416)

Table 3.11: Prediction of the average travel time with the GP emulator for levels 2, 3, 4 and 5. A zero-mean and RQ covariance functions are used in this model. The training set is formed by 256 points, and the number of KL coefficients retained is 18. The respective 95% confidence intervals for each of the predictions are shown in right column.

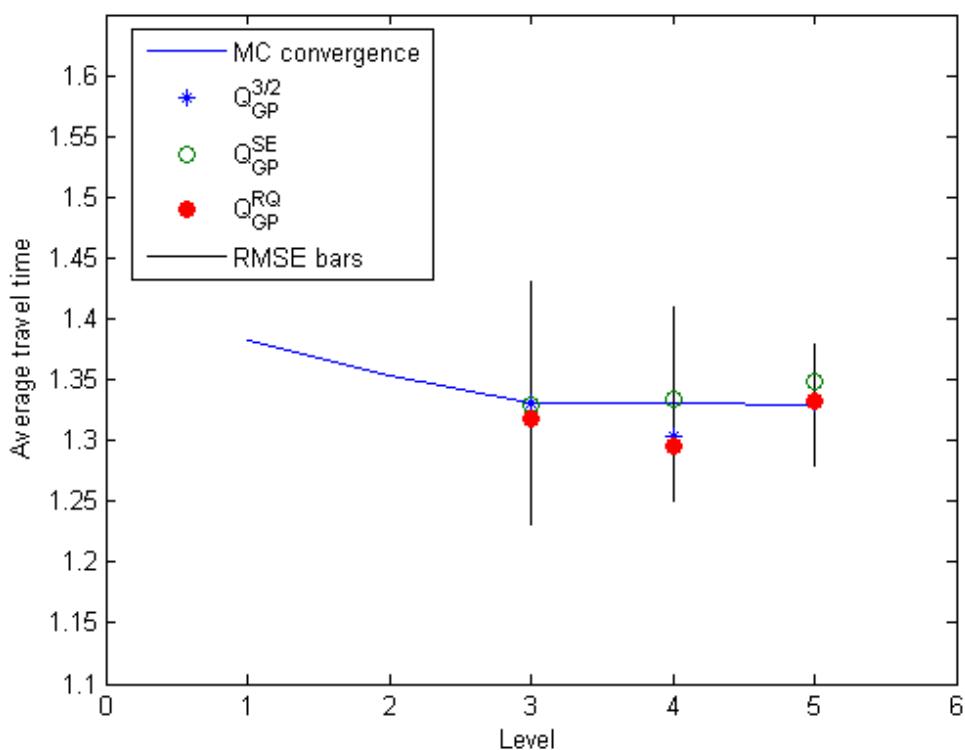


Figure 3.7: MC converge analysis (blue line) based of 10^5 travel time samples at each level (1-5). RMSE tolerances (black vertical bars) for levels 3-5: RMSE=0.1, RMSE= 0.08 and RMSE=0.05 respectively. T_{GP}^{SE} , $T_{GP}^{3/2}$ and T_{GP}^{RQ} are the predicted average travel time for each of the GP emulators.

Chapter 4

Application of GP emulation to the convectively-enhanced dissolution model

In previous chapters we presented some of the methodologies available to conduct UA in computer models. The simplicity of the model introduced in chapter 2 provided us with a deep understanding of the techniques used in terms of model implementation, testing and refinement. The goal of this chapter is to apply those modelling techniques to a more physical and computational complex real problem, namely, the convectively-enhanced dissolution process. In this case, the complexity of the simulator makes unfeasible to perform a full Monte Carlo UA and this strengthens the use of alternative methods as GP emulation.

The structure of this chapter will be analogous to the one followed in Chapter 2. We will start describing the proposed mathematical model for the physical problem, the parameters of interest, and the procedures followed to find and characterise numerical solutions, in other words, how we build a reliable simulator. Then, we will test and select the GP emulation model, use the GP emulator to make some predictions, and compare these results with approximations given by the MC method. In the final part of the chapter, we will conduct a full UA with the GP emulator and we will end this chapter with a comparison between

direct scalar GP emulation of the quantity of interest and GP field emulation plus simulation from the emulated field, as we did in Chapter 2.

4.1 The convectively-enhanced dissolution process

During CCS processes, the CO₂ is trapped in saline geological formations located deep underground. Once there, it reacts with minerals in the geologic formation, leading to the precipitation of secondary carbonate mineral (Ghesmat et al., 2011). When the CO₂ is dissolved into the brine, the density of the resulting solution is denser than the brine and this density difference from the new solute can result in instabilities, which one consequence may be plumes of CO₂ to grow downwards and form finger structures. The descending mixed fluid along the fingers will induce recirculation cells of brine fluid with an associated fluid entrainment into the fingers. The entrained brine reduces the density difference and the CO₂ concentration at the interface diffusive boundary layer, resulting in enhancement of the dissolution process (see e.g., Neufeld et al. (2010) and Ward et al. (2014)).

The dissolution flux will be characterized by a generalization of the Sherwood number (see, e.g., Ranganathan et al. (2012) and Ward et al. (2014)), which is a dimensionless measure of the convective flux across the upper boundary of the domain, and will be called the *surface flux (SF)*. We will consider the effect of the rock heterogeneity on the C-ED process in a porous medium, in which the solute undergoes a first order chemical reaction, by looking at how the *SF* is affected by rock heterogeneities. Thus, the main goal of this chapter will be to quantify the uncertainty we have in the amount of CO₂ dissolution flux that goes into the brine due to heterogeneity (or uncertainty) of the rock formation.

4.2 Mathematical model

We consider the dissolution of a solute (CO_2) in a fluid (brine) flowing in a two-dimensional heterogeneous, isotropic porous medium, with dimensions and orientation as shown in Figure 4.1. The solute locally increases the solution density but undergoes a first order reaction and turn into an inert product with no effect on the solution density.

The complex microscopic flow in a porous medium leads to an apparent macroscopic dispersive transport, adding to the molecular diffusion. We model this phenomenon by an apparent dispersion tensor, \mathbf{D} , depending on the local Darcy's velocity of the fluid \mathbf{u} (see e.g., Saffman (1959) or Scheidegger (1961)) as follows,

$$\mathbf{D} = D_m \mathbb{I} + \beta_T \|\mathbf{u}\| \mathbb{I} + (\beta_L - \beta_T) \frac{\mathbf{u} \otimes \mathbf{u}}{\|\mathbf{u}\|},$$

where \otimes represents the tensor product, \mathbb{I} is the unit (identity) tensor, D_m is the molecular diffusion coefficient of the solute in the considered fluid and β_L and β_T are respectively the longitudinal and transverse dispersion coefficient such that $\beta_L \geq \beta_T \geq 0$.

The governing equations defining the problem are described in the following section.

4.2.1 Governing equations

The equations used in this thesis to describe mathematically the C-ED process, are, the continuity equation (4.1), the Darcy's law¹ (4.2) and the convection-diffusion-reaction equation (4.3) (see, e.g., Ranganathan et al. (2012), Ward et al. (2014) and Xie et al. (2012)),

$$\frac{\partial \rho}{\partial t} + \nabla \cdot \rho \mathbf{u} = 0, \quad (4.1)$$

¹Note that in this chapter, Darcy's law is given in its general form and then, with further assumptions, we simplify it to $\nabla \cdot \mathbf{u} = 0$, while in Chapter 2, we considered initially an incompressible flow and therefore the density ρ was constant and the continuity equation presented by $\nabla \cdot \mathbf{q} = 0$.

$$\mathbf{u} = -\frac{K}{\mu}(\nabla P - \rho\mathbf{g}), \quad (4.2)$$

$$\phi \frac{\partial C}{\partial t} + \mathbf{u} \cdot \nabla C = \phi \nabla \cdot (\mathbf{D} \nabla C) - \gamma_c C, \quad (4.3)$$

where the variables C , $\mathbf{u} = (u_x, u_z)$ and P are the concentration of dissolved CO₂, the velocity in the x and z directions and the pressure respectively. The parameters K , μ , ϕ , γ_c and \mathbf{g} are the medium permeability field, fluid viscosity, rock porosity, reaction rate and acceleration due to gravity.

The density of the fluid is considered linear in the form, $\rho = \rho_0 + \beta_c C$, where ρ_0 and β_c are density of pure fluid and the volumetric expansion coefficient, and the change in density is assumed to be small, this later enables us to use the Boussinesq approximation (see e.g., Ward et al. (2014)), in which the density of the fluid is considered constant except in momentum source term, and the continuity equation is therefore reduced to the statement of a solenoidal velocity field that is satisfied by introducing a streamfunction formulation. With this consideration Equation (4.1) becomes:

$$\nabla \cdot \mathbf{u} = 0. \quad (4.4)$$

4.2.2 Coordinates system

We will consider a Cartesian coordinate system in two dimension where it is important to remark that the z-coordinate decreases in the same direction that gravity acts. Therefore (4.2) becomes:

$$\mathbf{u} = -\frac{K}{\mu}(\nabla P + \rho g \mathbf{e}_z), \quad (4.5)$$

where \mathbf{e}_z is the unitary vector corresponding to the ordinate axis.

4.2.3 Domain and boundary conditions

The dissolution process is considered in a porous material of depth $2H$ and length L (see Figure 4.1).

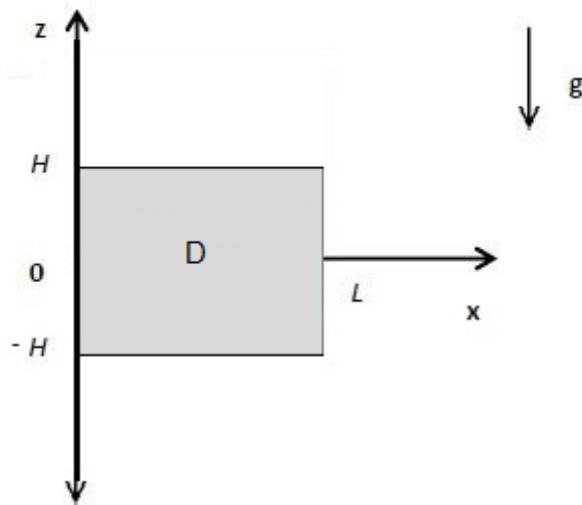


Figure 4.1: Two dimensional domain with horizontal length L and vertical depth $2H$, i.e., $D = [0, L] \times [-H, H]$. Gravity \mathbf{g} acts downwards along with the Z-axis.

The boundary conditions are:

Dirichlet:

$$C(x, z = H) = C_0$$

$$u_x(x = 0, z) = 0$$

$$u_x(x = L, z) = 0$$

$$u_z(x, z = \pm H) = 0$$

Neumann:

$$\begin{aligned}\frac{\partial C}{\partial z}(x, z = -H) &= 0 \\ \frac{\partial C}{\partial x}(x = 0, z) &= 0 \\ \frac{\partial C}{\partial x}(x = L, z) &= 0\end{aligned}$$

4.2.4 Streamfunction formulation

Equation (4.4) allows us to introduce the streamfunction, Ψ , where:

$$\mathbf{u}_x = \frac{\partial \Psi}{\partial z}, \quad \mathbf{u}_z = -\frac{\partial \Psi}{\partial x}$$

and therefore,

$$\frac{\partial \Psi}{\partial z} = -\frac{K}{\mu} \frac{\partial P}{\partial x}, \tag{4.6}$$

and

$$-\frac{\partial \Psi}{\partial x} = -\frac{K}{\mu} \left(\frac{\partial P}{\partial z} - (\rho_0 + \beta_c C)g \right) \tag{4.7}$$

So that, by differentiating (4.6) with respect to x , (4.7) with respect to z , and adding up both equations, the governing equations, (4.1), (4.2), (4.3), may be re-written in terms of the new unknowns, Ψ and C , as follows:

$$\frac{\partial}{\partial x} \left(\frac{1}{K} \frac{\partial \Psi}{\partial x} \right) + \frac{\partial}{\partial z} \left(\frac{1}{K} \frac{\partial \Psi}{\partial z} \right) + \frac{\beta_c g}{\mu} \frac{\partial C}{\partial x} = 0, \tag{4.8}$$

$$\phi \frac{\partial C}{\partial t} - \frac{\partial \Psi}{\partial z} \frac{\partial C}{\partial x} + \frac{\partial \Psi}{\partial x} \frac{\partial C}{\partial z} - \phi \nabla \cdot (\mathbf{D} \nabla C) + \gamma_c C = 0, \tag{4.9}$$

with boundary conditions:

Dirichlet:

$$\begin{aligned} C(x, z = H) &= C_0 \\ \Psi(x, z = \pm H) &= 0 \\ \Psi(x = 0, z) &= 0 \\ \Psi(x = L, z) &= 0 \end{aligned}$$

Neumann:

$$\begin{aligned} \frac{\partial C}{\partial z}(x, z = -H) &= 0 \\ \frac{\partial C}{\partial x}(x = 0, z) &= 0 \\ \frac{\partial C}{\partial x}(x = L, z) &= 0 \end{aligned}$$

4.2.5 Non-dimensional formulation

Equations (4.8) and (4.9) are non-dimensionalized using the following scalings:

Dimensionless variables:

$$(x', z') = \frac{(x, z)}{H}, \quad \Psi' = \frac{\Psi \mu}{HC_0 K_0 \beta_c g}, \quad C' = \frac{C}{C_0}, \quad t' = \frac{t C_0 K_0 \beta_c \rho}{\mu \phi H},$$

Dimensionless parameters:

$$(\beta'_L, \beta'_T) = \frac{(\beta_L, \beta_T) C_0 K_0 \beta_c g}{D_0 \mu}, \quad K' = \frac{K}{K_0}, \quad \mathcal{L} = \frac{L}{H},$$

Dimensionless numbers:

$$Ra = \frac{K_0 C_0 g \beta_c H}{\phi \mu D_0}, \quad Da = \frac{\gamma_c \mu H}{K_0 C_0 g \beta_c},$$

where β_T and β_L are respectively the longitudinal and transverse dispersion coefficient (see, e.g., Xie et al. (2012) and Hidalgo and Carrera (2009)), \mathcal{L} is the aspect ratio of the domain, and, Ra and Da , are respectively, the Rayleigh and Damköler numbers, related to the buoyancy driven flow and the ratio of the chemical reaction rate to the mass transfer rate (see, e.g., Ward et al. (2014)).

Substituting the new dimensionless variables, parameters and numbers in the equations (4.8) and (4.9), and dropping the primes for convenience, the governing equations and their boundary conditions stay as:

$$\frac{\partial}{\partial x} \left(\frac{1}{K} \frac{\partial \Psi}{\partial x} \right) + \frac{\partial}{\partial z} \left(\frac{1}{K} \frac{\partial \Psi}{\partial z} \right) + \frac{\partial C}{\partial x} = 0, \quad (4.10)$$

$$\frac{\partial C}{\partial t} - \frac{\partial \Psi}{\partial z} \frac{\partial C}{\partial x} + \frac{\partial \Psi}{\partial x} \frac{\partial C}{\partial z} - \frac{1}{Ra} \left(\frac{\partial J_x}{\partial x} + \frac{\partial J_z}{\partial z} \right) + DaC = 0, \quad (4.11)$$

where $\mathbf{J}=(J_x, J_z)$ is Fickian mass flux governed by Scheidegger-Bear's dispersion approach (Xie et al., 2012) corresponding to the term, $\mathbf{J}=\mathbf{D}\nabla C$, and the expressions for J_x and J_z are:

$$J_x = (1 + \beta_T \|\nabla \Psi\|_2) \frac{\partial C}{\partial x} + \frac{(\beta_L - \beta_T)}{\|\nabla \Psi\|_2} \left(\left(\frac{\partial \Psi}{\partial z} \right)^2 \frac{\partial C}{\partial x} - \frac{\partial \Psi}{\partial x} \frac{\partial \Psi}{\partial z} \frac{\partial C}{\partial z} \right) \quad (4.12)$$

$$J_z = (1 + \beta_T \|\nabla \Psi\|_2) \frac{\partial C}{\partial z} + \frac{(\beta_L - \beta_T)}{\|\nabla \Psi\|_2} \left(\left(\frac{\partial \Psi}{\partial x} \right)^2 \frac{\partial C}{\partial z} - \frac{\partial \Psi}{\partial x} \frac{\partial \Psi}{\partial z} \frac{\partial C}{\partial x} \right) \quad (4.13)$$

and

$$\|\nabla \Psi\|_2 = \left(\left(\frac{\partial \Psi}{\partial x} \right)^2 + \left(\frac{\partial \Psi}{\partial z} \right)^2 \right)^{1/2}$$

with boundary conditions:

Dirichlet:

$$\begin{aligned} C(x, z = 1) &= 1 \\ \Psi(x, z = \pm 1) &= 0 \\ \Psi(x = 0, z) &= 0 \\ \Psi(x = \mathcal{L}, z) &= 0 \end{aligned}$$

Neumann:

$$\begin{aligned} \frac{\partial C}{\partial z}(x, z = -1) &= 0 \\ \frac{\partial C}{\partial x}(x = 0, z) &= 0 \\ \frac{\partial C}{\partial x}(x = \mathcal{L}, z) &= 0 \end{aligned}$$

4.2.6 Quantity of interest

The quantity of interest for the C-ED problem will be the SF^1 introduced in Section 4.1. This quantity can be computed mathematically as follows:

If we integrate (4.11) over the domain D we have,

$$\int_D \left(\frac{\partial C}{\partial t} - \frac{\partial \Psi}{\partial z} \frac{\partial C}{\partial x} + \frac{\partial \Psi}{\partial x} \frac{\partial C}{\partial z} - \frac{1}{Ra} \left(\frac{\partial J_x}{\partial x} + \frac{\partial J_z}{\partial z} \right) + DaC \right) ds = 0, \quad (4.14)$$

if we now let $Mass = \int_D C ds$, we can write,

$$\frac{d}{dt} Mass - \int_D \left(\frac{\partial \Psi}{\partial z} \frac{\partial C}{\partial x} - \frac{\partial \Psi}{\partial x} \frac{\partial C}{\partial z} \right) ds = \frac{SF}{Ra} - Da \ Mass, \quad (4.15)$$

Since the second integral in the left hand side of the equation is zero² we can

¹ SF is defined here as a generalization of the Sherwood number since not only dispersion but also dispersivity is considered within the model (Ranganathan et al., 2012).

²Green's theorem states that if we let ∂D be a positively oriented, piecewise smooth, simple closed curve in a plane, and let D be the region bounded by ∂D . If P and Q are functions of (x, z) defined on an open region containing D and have continuous partial derivatives there,

re-write (4.15) as,

$$\frac{d}{dt} \text{Mass} = \frac{SF}{Ra} - Da \text{ Mass}, \quad (4.16)$$

with

$$SF = \int_D \left(\frac{\partial J_x}{\partial x} + \frac{\partial J_z}{\partial z} \right) ds. \quad (4.17)$$

Applying now the Divergence theorem to above integral and removing zero terms, we can finally write¹ SF as,

$$SF = - \int_0^L \left(1 + \beta_T \left| \frac{\partial \Psi}{\partial z} \right|_{z=1} \right) \left(\frac{\partial C}{\partial z} \right)_{z=1} dx. \quad (4.18)$$

It can be observed from equation (4.16), that the total mass of solute in the domain increases through solute injection across the upper boundary and decreases through the first order reaction. Note also that, in this particular problem, the more CO_2 is absorbed through the upper boundary, the more the process is considered to be efficient.

Although the mathematical formulation of the problem was described in terms of its transient formulation, in this thesis we are interested in the long term behaviour of the evolution of the problem and consequently we will consider only non-trivial solutions of the corresponding steady state condition. Under this

then the following equality holds, $\int_{\partial D} (Pdx + Qdz) = \int_D \left(\frac{\partial Q}{\partial x} - \frac{\partial P}{\partial z} \right) dx dz$. Thus, by taking $P = \frac{\partial \Psi}{\partial x} C$ and $Q = \frac{\partial \Psi}{\partial z} C$ and applying Green's theorem it follows that, $\int_D \left(\frac{\partial Q}{\partial x} - \frac{\partial P}{\partial z} \right) dx dz = \int_D \left(\frac{\partial \Psi}{\partial z} \frac{\partial C}{\partial x} - \frac{\partial \Psi}{\partial x} \frac{\partial C}{\partial z} \right)$, and so, $\int_{\partial D} (Pdx + Qdz) = \int_0^L Pdx + \int_{-H}^H Qdz - \int_L^0 Pdx + \int_H^{-H} Qdz = \int_0^L \frac{\partial \Psi}{\partial x} Cdx + \int_{-H}^H \frac{\partial \Psi}{\partial z} Cdz - \int_L^0 \frac{\partial \Psi}{\partial x} Cdx + \int_H^{-H} \frac{\partial \Psi}{\partial z} Cdz = 0$ (as the four latter integrals are all equal to zero).

¹Let $\mathbf{J} = (J_x, J_z)$, with J_x and J_z as described in (4.12) and (4.13) respectively. By Divergence theorem the follow equality holds, $\int_D (\nabla \cdot \mathbf{J}) dA = \int_{\partial D} \mathbf{J} \cdot \mathbf{n} ds$, where \mathbf{n} is the outward normal unitary vector to the boundary ∂D . By taking $\mathbf{n} ds = (dz, -dx)$, it follows that, $\int_{\partial D} \mathbf{J} \cdot \mathbf{n} ds = \int_{\partial D} J_x dz - J_z dx$, and thus, $\int_{\partial D} J_x dz - J_z dx = - \int_0^L J_z dx + \int_{-H}^H J_x dz + \int_L^0 J_z dx - \int_H^{-H} J_x dz$. Applying boundary conditions, $\int_{\partial D} J_x dz - J_z dx = - \int_0^L J_z dx = - \int_0^L (1 + \beta_T \left| \frac{\partial \Psi}{\partial z} \right|_{z=1}) \left(\frac{\partial C}{\partial z} \right)_{z=1} dx$. Note that $dx = 0$ on left and right boundaries and $dz = 0$ on bottom and top boundaries.

condition equation (4.16) reduces to:

$$SF = RaDa \text{ Mass}, \quad (4.19)$$

As Ra and Da are parameters of the model, and recalling that $\text{Mass} = \int_D C ds$, (4.19) shows that the SF is directly proportional to Mass , and therefore, for computing SF , it is enough with solving the equations (4.10) and (4.11) for Ψ and C , and then compute the corresponding integral for C . In the following section we will show how we find numerical solutions for the problem defined by equations (4.10) and (4.11).

4.2.7 Search of numerical solutions for the C-ED model

To solve the C-ED problem described in the previous section we use the finite element method (FEM) (see, e.g., Brenner and Scott (2002)). For the numerical implementation of the FEM we used the package AptoFEM¹.

The C-ED problem depends on several scalar parameters that need to be specified before attempting to find a numerical solution, namely, Ra , Da , β_L and β_T . The equations depend also on the permeability, K , which modeled as a random permeability field in the same way we showed in Section 2.8.4. Once these five parameters are specified, the equations (4.10) and (4.11) are solved for the unknown, namely, the streamfunction, Ψ , and the concentration, C . And finally, from C , we can compute the quantity of interest, the SF . Henceforth, throughout this chapter, we will refer to the simulator as a function, f , as follows:

$$f : \mathbf{K} \in \mathbb{R}^M \rightarrow SF \in \mathbb{R}. \quad (4.20)$$

The non-linearity of the system of equations leads to a multiple solutions problem, in other words, there exists more than one streamfunction and concentration scenarios, and therefore more than one SF , for a given permeability field. For instance, Ward et al. (2014) investigated a particular case of our problem,

¹AptoFEM is an in-house finite element (FE) code developed in the School of Mathematics at the University of Nottingham (Houston)

namely, $K=1$, $\beta_L=\beta_T=0$, and showed different types of solutions obtained by solving the equations with the same input parameters, Ra and Da .

The existence of multiple solutions (outputs) adds an additional challenge to the search of numerical solutions others than the trivial solution (which is called the no-flow solution). The FEM code is only able to find one solution (the no-flow) for the problem, and thus, to overcome this, it is necessary to use other techniques in conjunction with the FEM. In this thesis we use the Arclength continuation theory described in Cliffe et al. (2000a) as a tool for finding numerical solutions others than the no-flow solution. Figure 4.2 shows a representation of all the solutions found¹ with our scheme for the homogeneous C-ED problem and the respective scenario for the corresponding selection of different Ra . The reference value to characterise each computed solution represented in the Y-axis was the value of the streamfunction (velocity) at the center of the domain, according the corresponding Ra (X-axis). Note that before around² $Ra=42.5$, the bifurcation point, all the solutions found have constant concentration, we have called them no-flow solutions, and for Ra greater than around $Ra=42.5$ there are three different solutions. (For more details about bifurcation problems, see, e.g., Cliffe et al. (2000a) and Ward et al. (2014)).

To test and validate the simulator, for the no-flow solution comparison, we used the analytical solution for the homogeneous case, $K = 1$, provided in Ward et al. (2014), namely,

$$C_0(x, z) = \frac{\cosh(\sqrt{RaDa}(1 - z))}{\cosh(2\sqrt{RaDa})}, \quad \Psi_0(x, z) = 0. \quad (4.21)$$

For the non-trivial solutions, we compared our results with the results reported in Ward et al. (2014) for the cases of homogeneous porous medium ($K = 1$ in this case), which were obtained with a different numerical approach (the spectral method). We used the RE to compare the results and all the comparisons showed a RE of at least 10^{-5} .

¹It is important to remark that these correspond to numerical solutions we found but it does not mean that they are the only solutions for those inputs.

²We do not have an exact value for the bifurcation point. The fact that we found different branches of solutions around that value shows us its existence.

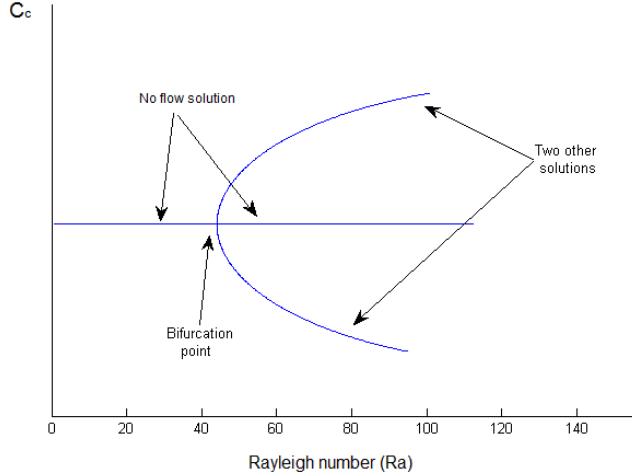


Figure 4.2: Bifurcation diagram with respect to Ra . Blue lines shows steady-state solutions projected on the streamfunction value at the center of the domain, C_c . The other model parameters are $K = 1$, $Da=0.1$ and $\beta_T = \beta_L = 0$.

4.2.7.1 Arc length continuation for finding numerical solutions

The steps followed to find numerical solutions of the C-ED model with Arc length continuation approach are,

1. Perturb equation (4.10) with small $\varepsilon > 0$ leading to:

$$\frac{\partial}{\partial x} \left(\frac{1}{K} \frac{\partial \Psi}{\partial x} \right) + \frac{\partial}{\partial z} \left(\frac{1}{K} \frac{\partial \Psi}{\partial z} \right) + \frac{\partial C}{\partial x} + \varepsilon = 0, \quad (4.22)$$

2. Consider the redefined system of equations formed by (4.22) and (4.11), and solve for different¹ Ra .

The sketch of the solutions for continuation paths (Cliffe et al., 2000a) with

¹Suppose we want to find a numerical approximation of Ψ and C for $Ra = 21$ following Arc length continuation approach, then, if for instance we use Newton's method as numerical scheme, we need the increments of the new desired Ra to be small enough to guarantee that the new continuation solution is within the same path (see red line in figure 4.3). This can be obtained by using the solution obtained for $Ra = 20$ as initial guess in Newton's method.

respect to Ra are showed in figure 4.3 (blue and red lines).

3. Once reached the desired Ra , start reducing¹ the perturbation $\varepsilon > 0$ till set $\varepsilon = 0$ which corresponds to the non-trivial (solutions distinct of no-flow solution, i.e. $\Psi = 0$) solution for the original problem for the desired Ra .

Figure 4.3 shows representation of the solutions (red lines) for the perturbed problem and the numerical solutions corresponding to the original problem (blue lines).

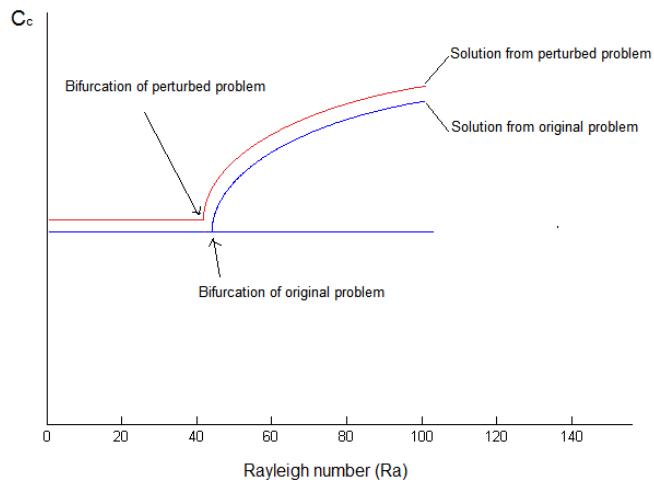


Figure 4.3: Bifurcation diagram with respect to Ra and $\varepsilon > 0$. Blue lines shows steady-state solutions projected on the concentration value at the center of the domain, C_c . The other parameters of the model are $Da=0.1$, $\beta_L = \pi/2$ and $\beta_T = \beta_L/10$.

As we showed in Section 2.8.5, the equations (4.10) and (4.11) becomes a new system of PDEs with random inputs, $K(x, z, \omega)$, which is solved for the stream-function, $\Psi(x, z, \omega)$, and the concentration, $C(x, z, \omega)$, which become also random fields. For simplicity, in the following we will omit the random term ω .

¹This is done exactly in the same way that we do for Ra , i.e., in Point 2 we do continuation on Ra and in Point 3 we do continuation on ε .

Once introduced the C-ED problem and the simulator, we will show next some examples of the streamfunction and concentration fields obtained for given a heterogeneous permeability fields.

4.2.8 Solutions for the C-ED problem

In this section we show some examples of the numerical solutions found with the simulator for the C-ED problem with parameters $Ra=60$, $Da=0.1$, $\beta_L = \pi/2$ and $\beta_T = \beta_L/10$. The permeability fields used as inputs in the simulator are generated in the same way we discussed in Section 2.8.5.

Figures 4.5 and 4.6 show the two non-trivial solutions, i.e., the solution at the upper branch and the solution at the lower branch, represented through red circles in figure 4.4. The figures show the contours of streamfunction, Ψ , and concentration, C , for the same permeability field, K .

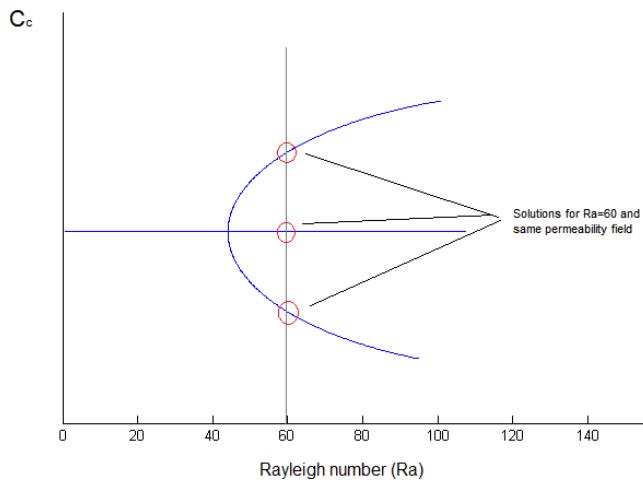


Figure 4.4: Illustrative representation of the bifurcation scenery against the Ra for a given heterogeneous permeability field. The red circles represents three different numerical solutions of the C-ED problem with parameters: $Ra=60$, $Da=0.1$, $\beta_L = \pi/2$ and $\beta_T = \beta_L/10$. The blue horizontal line represents the no-flow solutions branch.

To finish with the analysis of particular numerical solutions of the C-ED problem, we will investigate how spacial variations of the permeability field affect the SF .

4.2.9 Impact of the rock heterogeneity on the SF

In order to investigate how the heterogeneity affects the SF , for a given discrete heterogeneous permeability field, $\mathbf{K}_i \in \mathbb{R}^M$, we find the average value of the field, $\bar{\mathbf{K}}_i$, and compute the corresponding SF for this homogeneous case.

We have selected four cases in which the heterogeneity of the permeability affects drastically the values of the SF when compared with the homogeneous case. The dimensionless domain for these example will be $D = [0, \frac{\pi}{2}] \times [-1, 1]$. The rest of the parameters in equations (4.10) and (4.11) will be: $Ra = 100$, $Da = 0.1$, $\beta_L = \frac{\pi}{2}$ and $\beta_T = \frac{\beta_L}{10}$. Under this conditions, the SF for the no-flow solution, SF_0 , which is independent of the value of the permeability field, has a value of $SF_0 = 4.97$, which is used as a reference value in the results.

Figure 4.7 shows an averaged permeability of $\bar{\mathbf{K}}_1 = 0.86$, with relatively low values of the permeability around all the boundary, and high values of the permeability at the central region. Under this condition, the SF for the heterogeneous field, $SF_{\mathbf{K}_1} = 5.17$, is lower than the one obtained for the homogeneous case, $SF_{\bar{\mathbf{K}}_1} = 5.42$. Although in this case a clear region of relative larger permeability is observed along one of the main diagonals of the domain, the heterogeneity structure of the permeability field affects the one cell solution of the problem resulting in a reduction of the surface flux.

Figure 4.8 shows an averaged permeability of $\bar{\mathbf{K}}_2 = 0.61$, and the corresponding solution of the homogeneous case is given by the no-flow solution, $SF_0 = 4.97$. In contrast with the previous case, here the permeability shows larger permeability values at the bottom left and top right corners, and lower values at the central region where the we have almost a constant permeability. In this case, the existence of this lower permeability values channel leads to the formation of a weak plume flow, not present in the homogeneous case, this fact increases the flux, $SF_{\bar{\mathbf{K}}_2} = 4.97$, up to $SF_{\mathbf{K}_2} = 5.01$. Figure 4.8(e) shows the existence of two cells

in the streamfunction contours, a dominant cell along the main diagonal region with lower permeability values and a secondary small recirculation cell at the top left corner region with larger permeability values.

Figure 4.9 presents an averaged permeability of $\bar{K}_3 = 0.78$. In this case, there is a dominant region with low permeability values in the right half of the domain, which leads to the original two cells solution, corresponding to the homogeneous case, to pass into a single cell solution (occupying this low permeability region) and a resulting reduction in the SF from 5.58 (homogeneous case) to 5.00 (heterogeneous case), which is very close to the value of the no-flow solution.

Finally in Figure 4.10 an example with an averaged permeability of $\bar{K}_4 = 0.66$ is considered. Figure 4.10(a) shows isolated pokes of low and high permeability values with larger values at the top boundary layer of the domain. In this example, the two cases, homogeneous and heterogeneous, reveal the existence of two recirculation zones moving in opposite direction, although the resulting dissolution fluxes, $SF_{\bar{K}_4} = 5.03$ and $SF_{K_4} = 5.00$, are very similar.

From these results, we can deduce that the magnitude of the local values of the permeability does not affect drastically the changes in the SF , whereas the existence of “channels” of constant permeability values connecting different regions of the domain seems to impact on the magnitude of the resulting surface flux.

In the following section we will use all the tools learnt in Chapter 2 in order to build the GP emulator for the C-ED simulator.

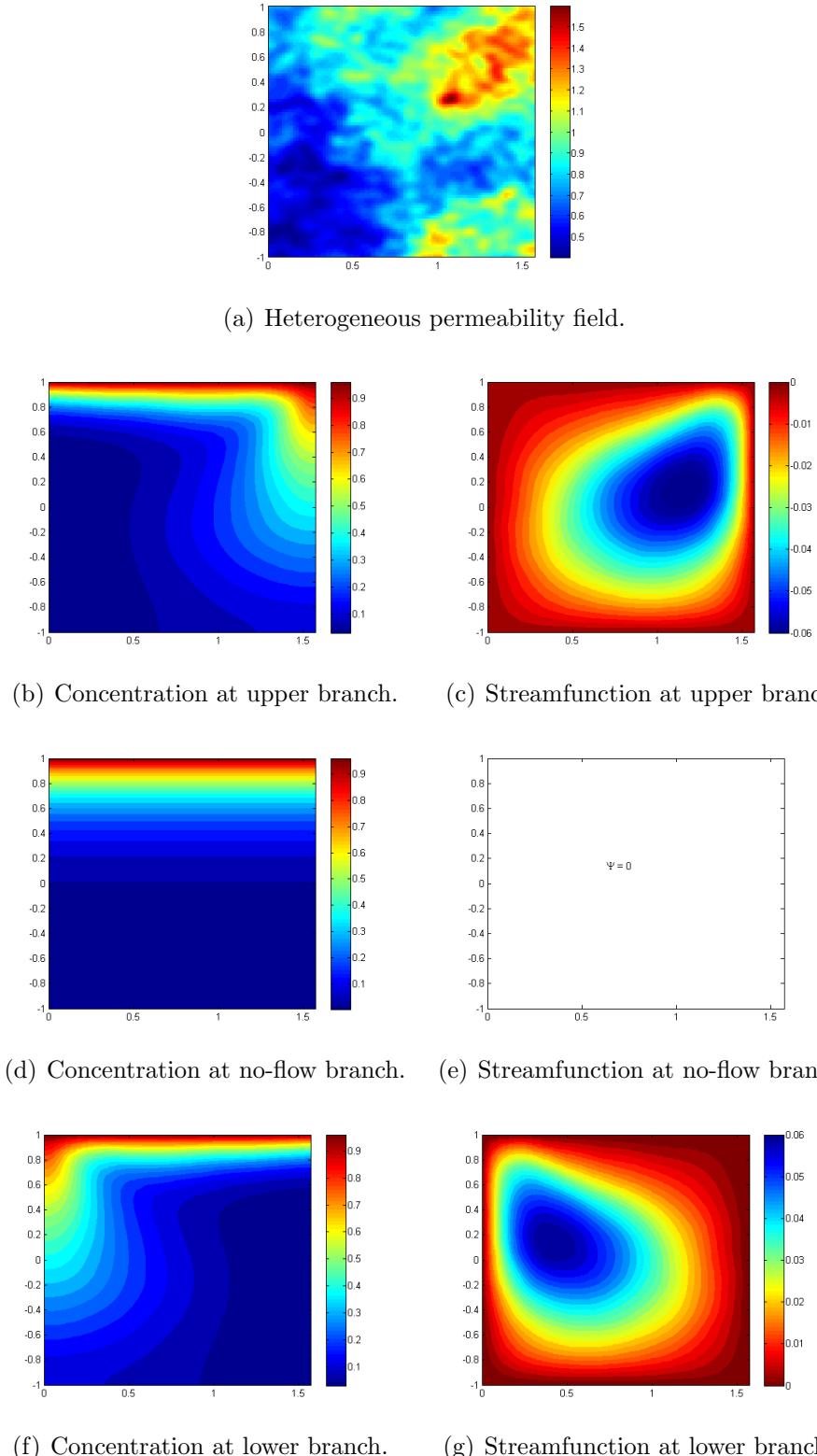


Figure 4.5: Contours of the concentration (left) and streamfunction (right) for a given permeability field (top) for the C-ED problem with parameters: $Ra=60$, $Da=0.1$, $\beta_L = \pi/2$ and $\beta_T = \beta_L/10$.

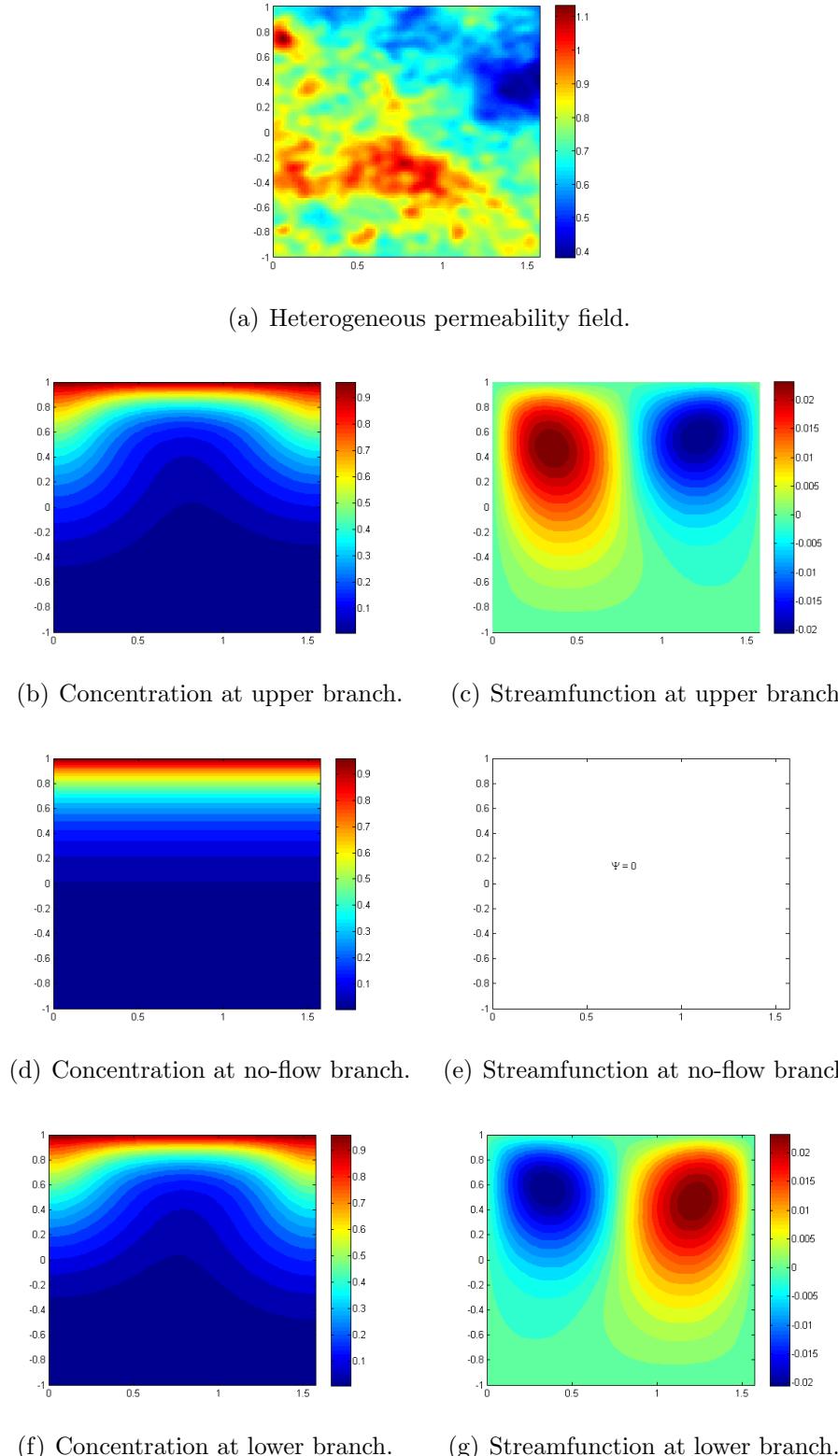


Figure 4.6: Contours of the concentration (left) and streamfunction (right) for a given permeability field (top) for the C-ED problem with parameters: $Ra=60$, $Da=0.1$, $\beta_L = \pi/2$ and $\beta_T = \beta_L/10$.

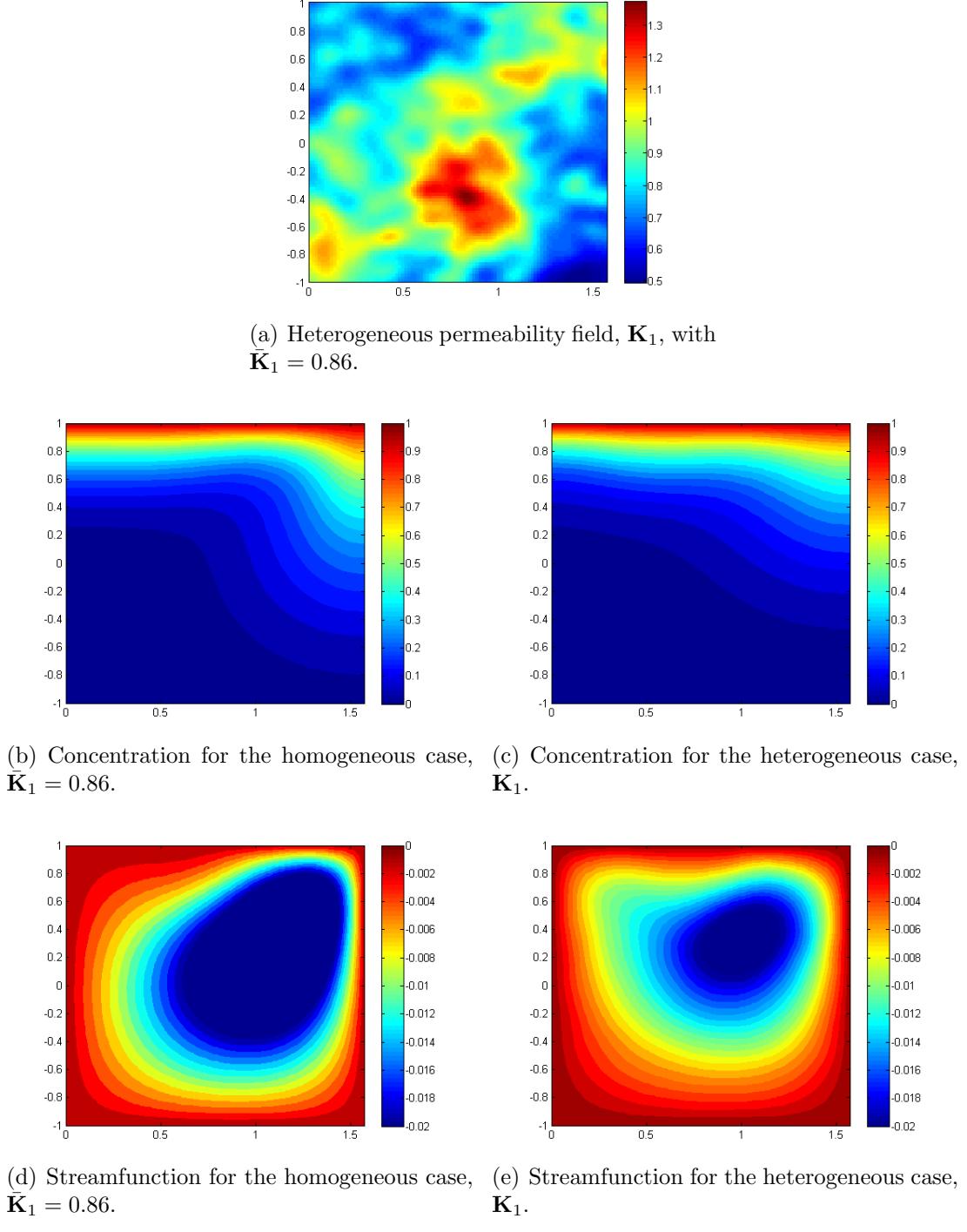


Figure 4.7: Concentration and Streamfunction contours for the homogeneous, $\bar{\mathbf{K}}_1 = 0.86$, and heterogeneous, \mathbf{K}_1 , permeability fields for the C-ED problem with parameters: $Ra=100$, $Da=0.1$, $\beta_L = \pi/2$ and $\beta_T = \beta_L/10$. The corresponding SFs for the homogeneous and heterogeneous cases, respectively, are $SF = 5.42$ and $SF = 5.17$.

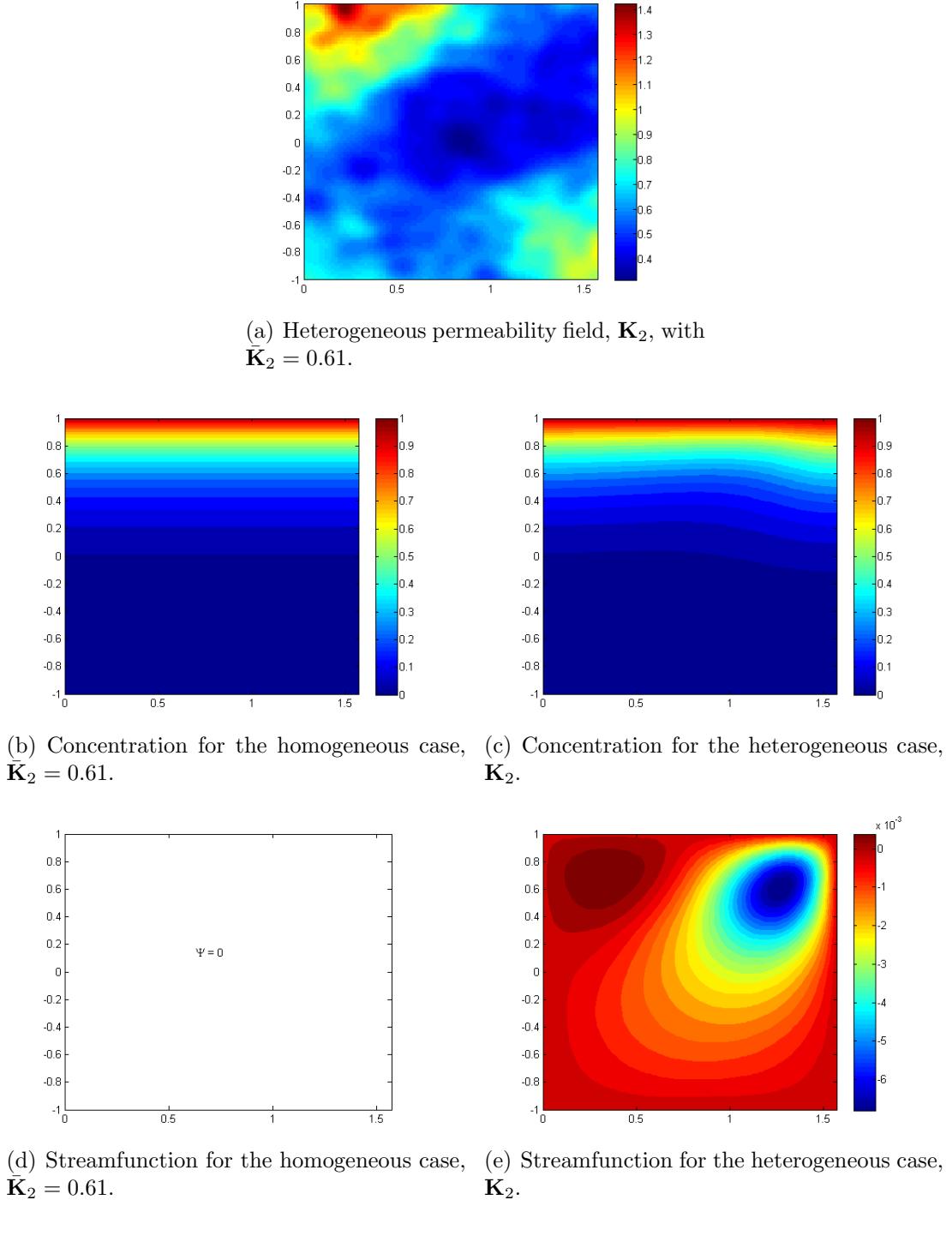


Figure 4.8: Concentration and Streamfunction contours for the homogeneous, $\bar{\mathbf{K}}_2 = 0.61$, and heterogeneous, \mathbf{K}_2 , permeability fields for the C-ED problem with parameters: $Ra=100$, $Da=0.1$, $\beta_L = \pi/2$ and $\beta_T = \beta_L/10$. The corresponding SFs for the homogeneous and heterogeneous cases, respectively, are $SF = 4.97$ and $SF = 5.01$.

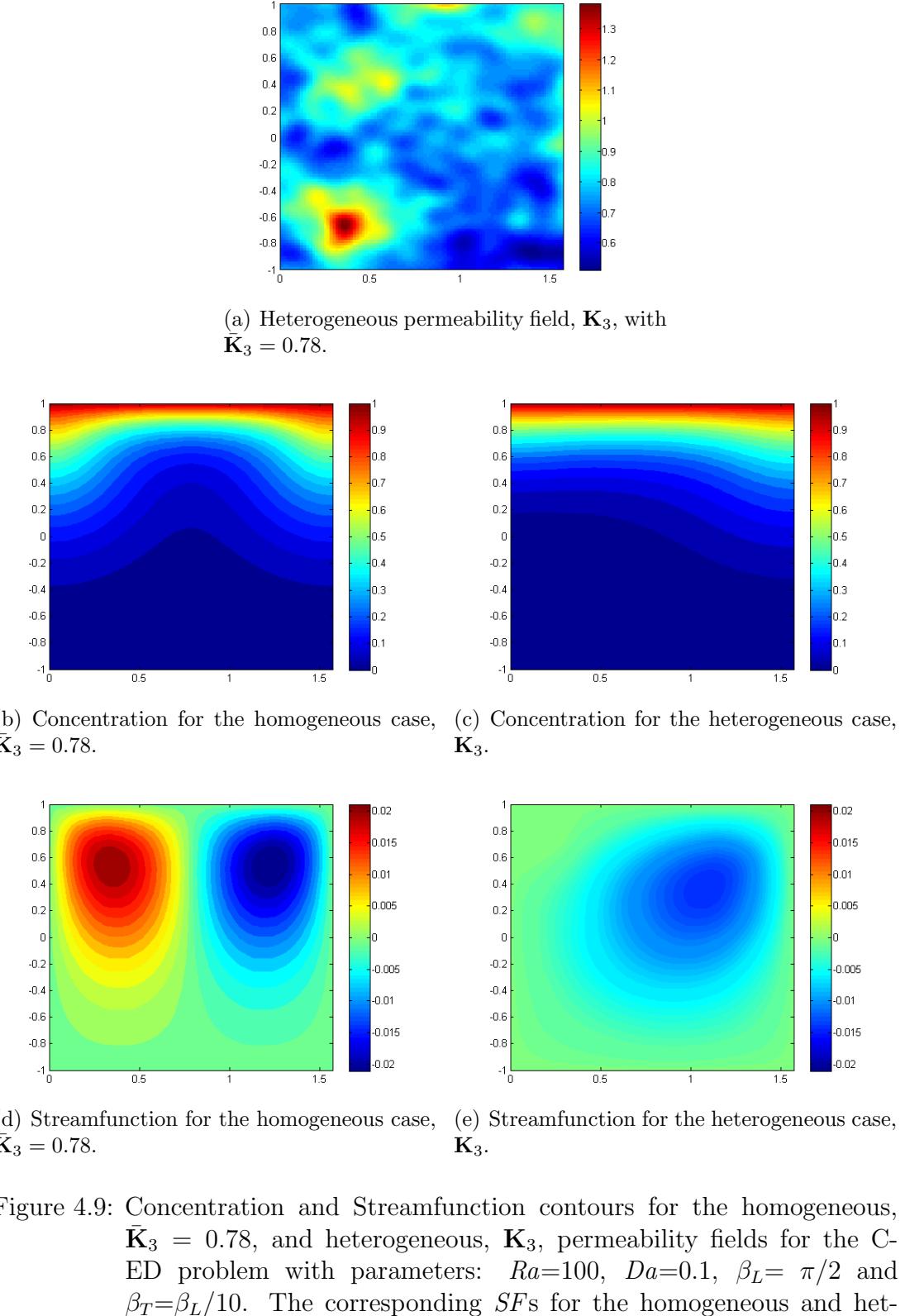
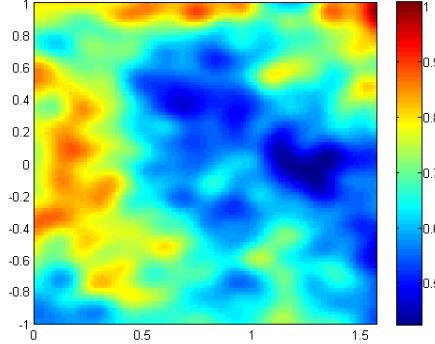
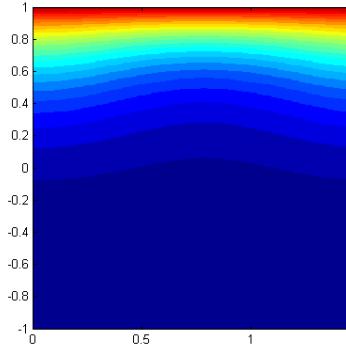


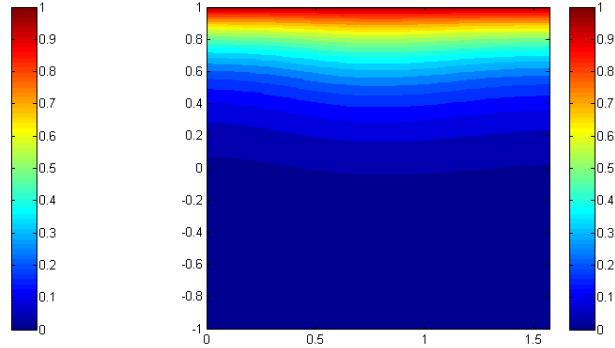
Figure 4.9: Concentration and Streamfunction contours for the homogeneous, $\bar{\mathbf{K}}_3 = 0.78$, and heterogeneous, \mathbf{K}_3 , permeability fields for the C-ED problem with parameters: $Ra=100$, $Da=0.1$, $\beta_L = \pi/2$ and $\beta_T = \beta_L/10$. The corresponding SFs for the homogeneous and heterogeneous cases, respectively, are $SF = 5.58$ and $SF = 5.00$.



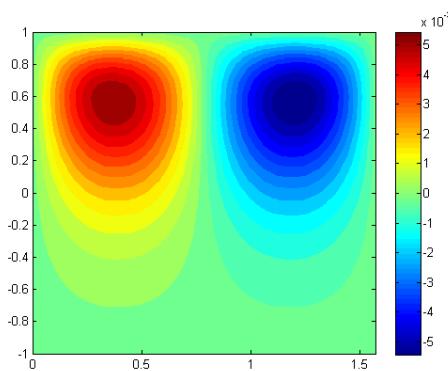
(a) Heterogeneous permeability field, \mathbf{K}_4 , with $\bar{\mathbf{K}}_4 = 0.66$.



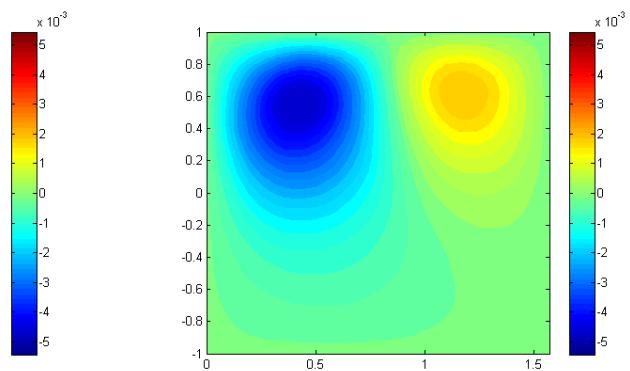
(b) Concentration for the homogeneous case, $\bar{\mathbf{K}}_4 = 0.66$.



(c) Concentration for the heterogeneous case, $\bar{\mathbf{K}}_4$.



(d) Streamfunction for the homogeneous case, $\bar{\mathbf{K}}_4 = 0.66$.



(e) Streamfunction for the heterogeneous case, $\bar{\mathbf{K}}_4$.

Figure 4.10: Concentration and Streamfunction contours for the homogeneous, $\bar{\mathbf{K}}_4 = 0.66$, and heterogeneous, \mathbf{K}_4 , permeability fields for the C-ED problem with parameters: $Ra=100$, $Da=0.1$, $\beta_L = \pi/2$ and $\beta_T = \beta_L/10$. The corresponding SFs for the homogeneous and heterogeneous cases, respectively, are $SF = 5.03$ and $SF = 5.00$.

4.3 GP emulator for the C-ED problem

As we discussed in the previous section, the C-ED model depends on several parameters, namely, Ra , Da , β_L and β_T , which have to be chosen before running the simulator to compute the corresponding SF . In this thesis we will be interested in analysing two scenarios, first, what happens “close” to a bifurcation point? and second, what happens “far” from a bifurcation point?. As the bifurcation point for the homogeneous case (see Figure 4.2) is located in the vicinity of $Ra = 42.5$, we have selected two cases of study, $Ra = 60$ (close to the bifurcation) and $Ra=100$ (far from the bifurcation). The rest of the parameters will remain fixed as, $Da = 0.1$, $\beta_L = \pi/2$, and $\beta_T = \beta_L/10$.

In both cases, $Ra = 60$ and $Ra = 100$, for a given permeability field, the simulator returns an output (SF) that can be either a trivial (no-flow solution) or a non-trivial solution. The SF arisen from a no-flow solution will be denoted by SF_0 . In the following sections, we will restrict ourselves to the study of non-trivial solutions, and later in the chapter, we will consider the whole set of possibilities. For doing this, we run the simulator for the usual 256 design points, ξ_i , and compute the corresponding observed SF_i to form the training set. Then, we remove all the pairs, $(\xi_i, f(\xi_i)) = SF_0$, and consider the rest of the set as the new training set for the GP emulator.

Let $f(\cdot)$ represent the simulator described in (4.20), which takes as input a random permeability field and return the corresponding surface flux, SF . For building a GP emulator for the simulator $f(\cdot)$, we will follow these steps:

1. Choose the appropriate set of design points, ξ_i , where to run the simulator.
2. Form the corresponding random permeability fields, \mathbf{K}_i , associated with the design points ξ_i .
3. Run the simulator, $f(\cdot)$, at those \mathbf{K}_i and obtain the corresponding SF_i in order to form the training set \mathcal{D} .
4. Remove all trivial solutions from the training set.

5. Use GP regression described in Section 1.3.3 to approximate the entire function $f(\cdot)$ with $\hat{f}(\cdot)$.

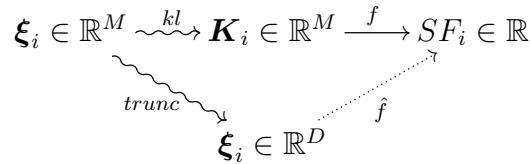


Figure 4.11: Emulation diagram. Above, the set of design points, $\xi_i = (\xi_1, \dots, \xi_D, \dots, \xi_M)$, used to generate the log Gaussian RF, \mathbf{K}_i , with a KL decomposition. These \mathbf{K}_i are then used to compute the corresponding surface fluxes, SF_i , with the simulator f . These SF_i are then be used as observed values in the training set. Below, the set of truncated design points, $\xi_i = (\xi_1, \xi_2, \dots, \xi_D)$, to form the training set along with the observed surface fluxes, SF_i . \hat{f} represents the GP emulator which is able to predict the surface flux, $SF_i^* \in \mathbb{R}$, for a given test case $\xi_i^* \in \mathbb{R}^D$.

Let us now apply the LOO-CV methodology described in Section 2.6, and used in Section 2.9.1 for the travel time simulator, to decide the structure of our C-ED GP emulator, and proceed with the UQ of the CDF of the SF for each of the two cases, $Ra = 60$ and $Ra = 100$.

Case $Ra=60$

Following the same strategy used in Chapter 2, we apply the LOO-CV method to a training set of 174 points (number of points after removing trivial solutions), and look for the GP emulator which its predicted data lie within the 95% CI 95% of the time. In forthcoming sections, we will only show plots of the selected GP model.

After testing our GP emulator with a zero-mean function and the four covariance functions proposed in Section 2.2, we obtained the following results: The percentage of points out of the range given by the LOO-CV method for the models using the SE , $Matérn_{\frac{3}{2}}$, $Matérn_{\frac{5}{2}}$, and RQ covariance functions is, respectively,

5.04, 4.02, 4.64 and 4.80. According to this results, we will build our GP emulator based upon a zero-mean and $\text{Matérn}_{\frac{3}{2}}$ covariance functions.

Figure 4.12 shows the predicted values (red) and 95% confidence intervals (black vertical bars) for each of the predictions, along with the observed values (blue) for each of the GP models. The percentage of points out of the range for the $\text{Matérn}_{\frac{3}{2}}$ model is 4.02% and so, 95.98% of the predictions are within the 95% acceptance interval.

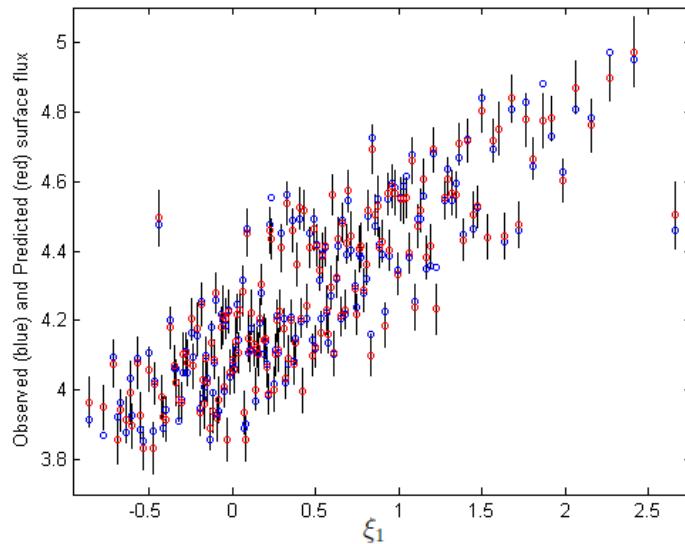


Figure 4.12: Predicted values (red) and their 95% bounds (black bars) given by the emulator against observed values (blue) for the LOO-CV from a design of 174 points and using a mean-zero function and a $\text{Matérn}_{\frac{3}{2}}$ covariance function. 4.02% of the observed values out off range.

To choose the optimum number of KL coefficients retained in this model, we will use the *NLML* (2.14), *DS* (2.15) and *MSE* (2.16).

Figure 4.13 shows the different scores plotted against the number of KL coefficients retained. In this case, all the scores follow the same decreasing tendency when increasing the number of KL coefficients used. The plots show that after around 14 KL coefficients, the scores are not changing significantly. Therefore, it seems sensible to build our training set based on 14-dimensional design points, i.e., $D = 14$ in the emulation diagram 4.11.

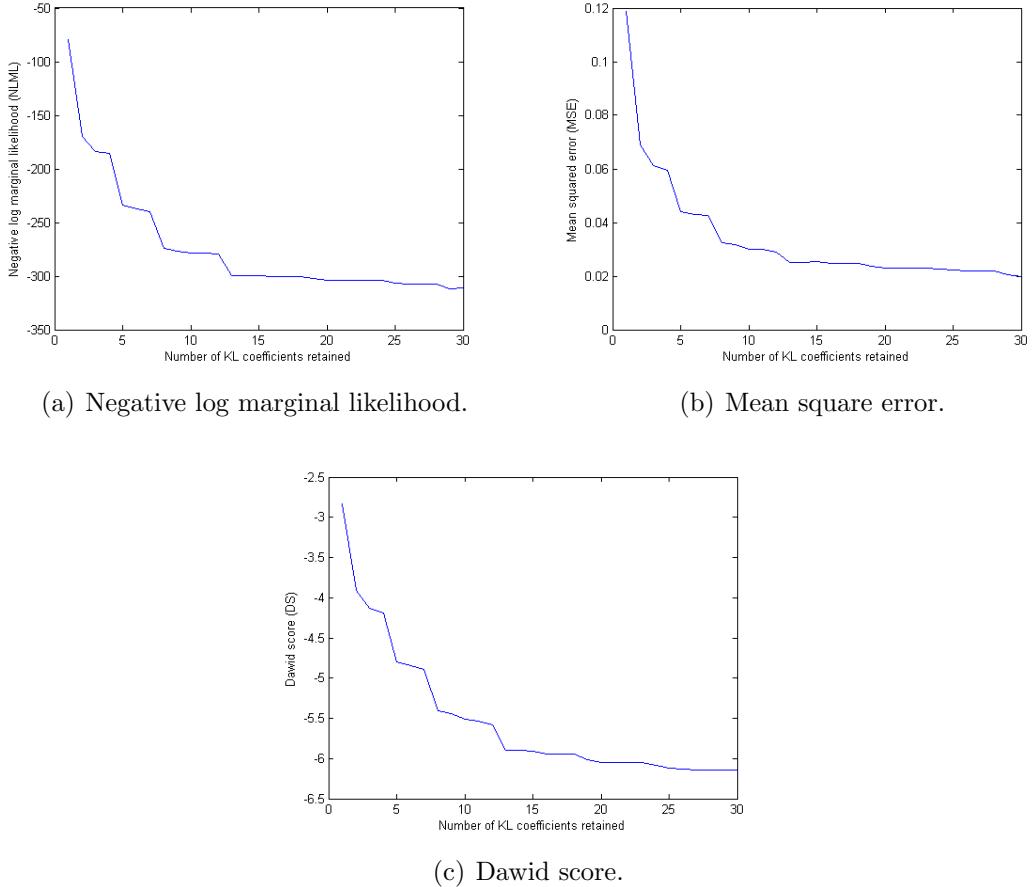


Figure 4.13: Different scores for a GP emulator built with a mean-zero function, $\text{Matérn}_{\frac{3}{2}}$ covariance function and 174 design points. The parameters of the C-ED simulator are: $Ra = 60$, $Da = 0.1$, $\beta_L = \pi/2$, and $\beta_T = \beta_L/10$. The x-axis represents the number of KL coefficients retained for each score.

As in Section 2.6, we show in Figure 4.14 the scatterplot between *observed SF = predicted SF* for this GP emulator. We also compute the relative error between 1000 observed surface fluxes and the corresponding predictions by running our GP emulator with $\{1\}$, $\{1, 2\} \dots \{1, 2, \dots, 174\}$ design points (i.e., we run the emulator 174 times). Figure 4.15 suggests that the adequate number of design points for this problem could be around 32 (power of 2), as taking more points would not improve significantly the GP emulator performance (measured in terms of the RE). This latter, becomes extremely important for this simulator since a single run for one point of the training set can last up to 20 minutes. Thus, in the following, and for this case, $Ra = 60$, we will use only the first 32 design points in our training set.

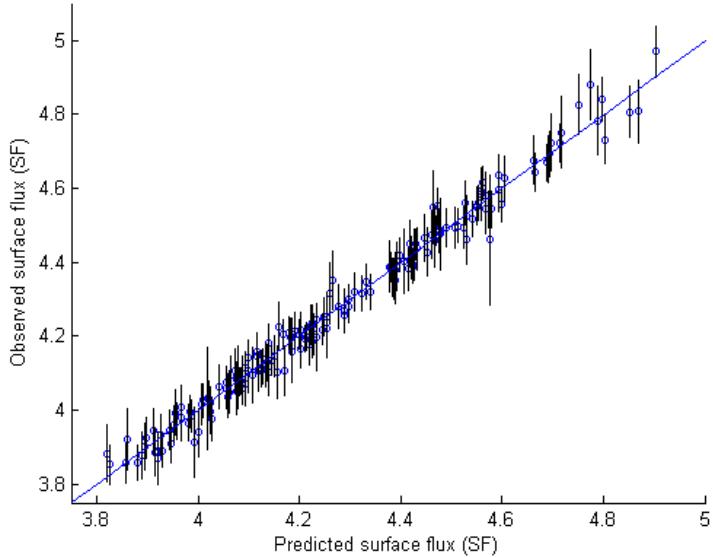


Figure 4.14: Observed and predicted surface fluxes for a design of 174 points, mean-zero function, and $Matérn_{\frac{3}{2}}$ covariance function. The solid line shows *observed SF = predicted SF*.

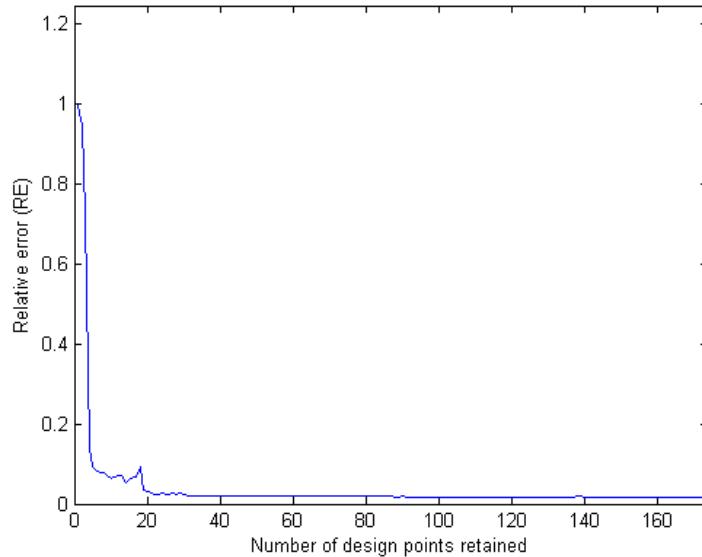


Figure 4.15: Relative error between 1000 observed and predicted surface fluxes against the number of design points retained using the $\text{Matérn}_{\frac{3}{2}}$ covariance function. The curve shows a decreasing tendency and it shows that after around 32 design points retained there is no variation in the relative error.

4.4 Uncertainty distribution of the *SF*. Case $Ra=60$

Due to the computational complexity of the C-ED simulator, it is not feasible¹ to use MC method to find the uncertainty distribution of the *SF*, and this, precisely enhances the use of GP emulation as an alternative to MC.

Before predicting the CDF of the *SF* with our GP emulator, we will use the MC method, based on 1000 samples, to show an estimation of the CDF of the

¹Actually, it takes 8 full days to run this simulator 1000 times. So, a full MC UA, which requires around 10^5 runs, would take approximately 800 days.

SF , and thus, have an idea of what type of CDF we expect. As in Chapter 2, we use the MC method described in Section 1.2.2 to approximate that CDF. To do that, we take $T = SF$ and $\mathbf{X}_M = \mathbf{K}$, then $T_M^{(i)} = SF_i$. We approximate the CDF by the ECDF with a sample of 1000 SF_i . To compute the ECDF of SF_i we follow the same procedure discussed in Section 2.10.1.

Figure 4.16 shows the ECDF of the SF based on the MC method with a sample of size 1000. The blue line is the estimation of the CDF of SF and the dashed lines the 95% uncertainty bounds for this empirical distribution.

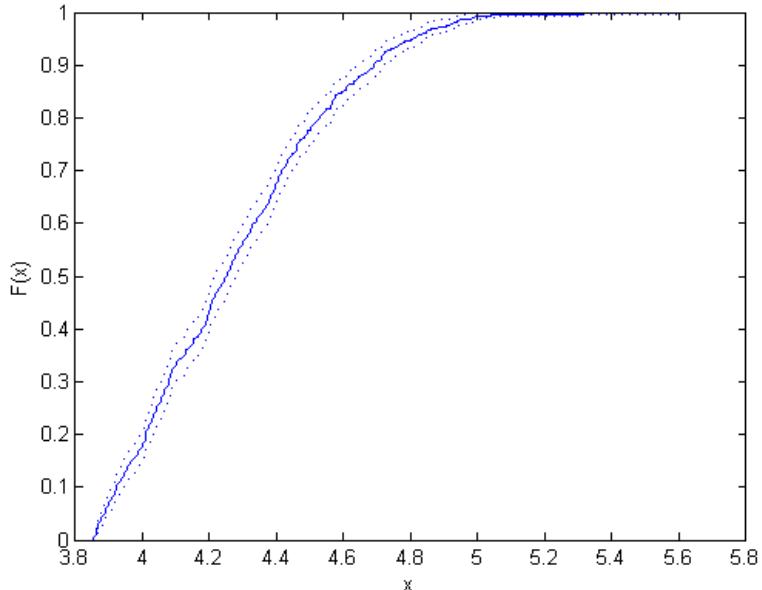


Figure 4.16: Monte Carlo ECDF (black line) based on 1000 surface fluxes for the case $\text{Ra}=60$. The dashed lines show the 95% uncertainty bounds.

In the following section we will use the GP emulator to perform a full UA of the distribution of the surface flux.

4.4.1 Using Gaussian process emulation to predict the cumulative distribution function of the surface flux

In this section, we will apply the same procedure described in Section 2.10.1 to quantify the uncertainty distribution of the surface flux for the case $Ra = 60$.

Figure 4.17 shows the ECDF (black) based on 1,000 samples of the surface flux computed with the MC method and GP posterior samples (green). Figure 4.18 shows the 2.5th and 97.5th percentiles (dashed magenta), the GP posterior mean (red) of the cumulative distribution function, and MC ECDF (black line).

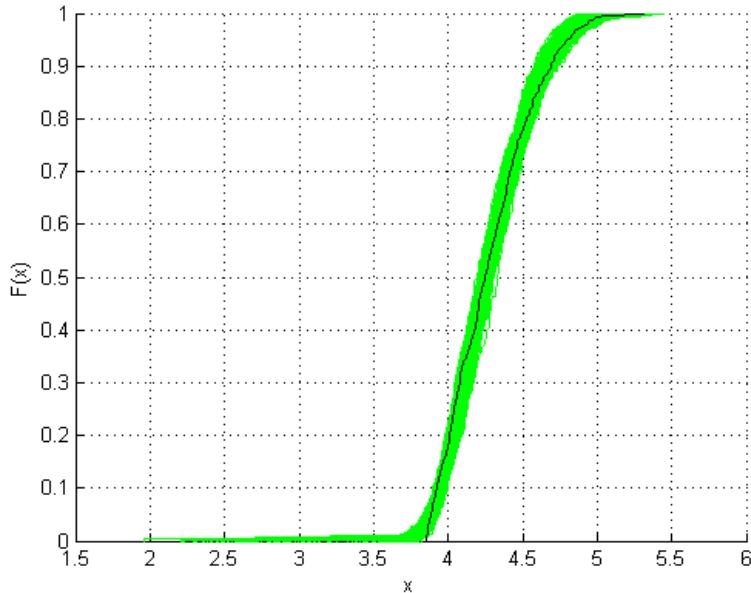


Figure 4.17: GP Posterior ECDFs samples, F_i , approximating the MC estimation of the CDF of the travel time of a convected particle in groundwater flow. The GP emulator prior specifications are: mean-zero function, SE covariance function, 174 design points and 14 KL coefficients retained.

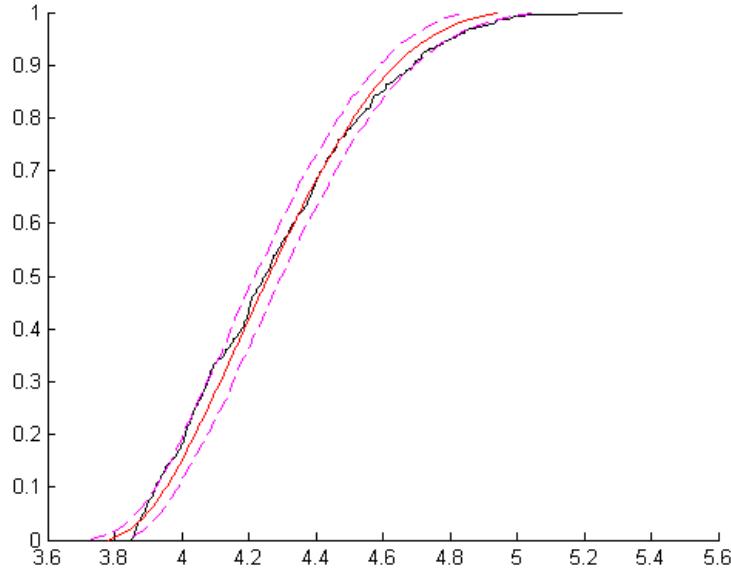


Figure 4.18: GP uncertainty analysis of the CDF of the travel time of a convected particle in groundwater flow. The GP emulator prior specifications are: mean-zero function, SE covariance function, 174 design points and 14 KL coefficients retained.

Case $Ra=100$

In this case, the number of design points removed from the initial training set of 256 points is 64, and thus, the training set for this GP emulator is formed by 192 points.

The percentage of points out of the range given by the LOO-CV methodology for the models using the SE , $Matérn_{\frac{3}{2}}$, $Matérn_{\frac{5}{2}}$, and RQ covariance functions is, respectively, 5.32, 4.22, 4.42 and 4.06. According to this results, we will build our GP emulator based upon a zero-mean and RQ covariance functions.

Figure 4.19 shows the predicted values (red) and 95% confidence intervals (black vertical bars) for each of the predictions, along with the observed values (blue) for each of the GP models. The percentage of points out of the range for the $Matérn_{\frac{3}{2}}$ model is 4.06% and so, 95.94% of the predictions are within the 95% acceptance interval.

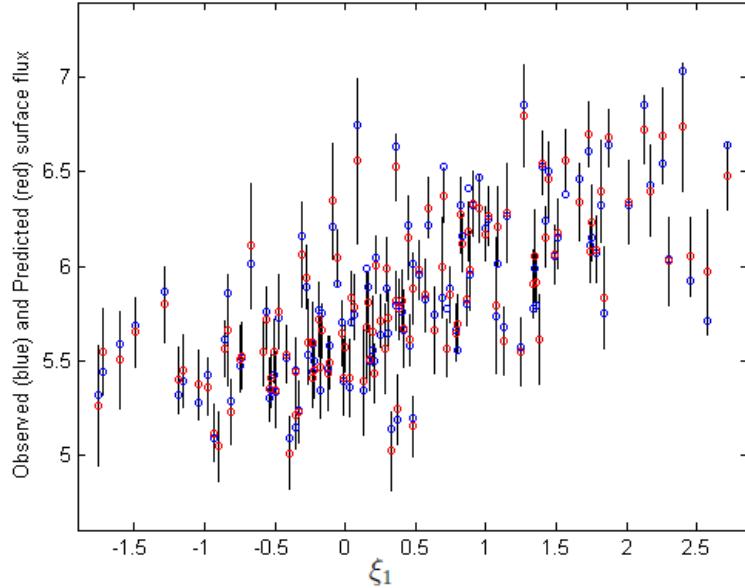


Figure 4.19: Predicted values (red) and their 95% bounds (black bars) given by the emulator against observed values (blue) for the LOO-CV from a design of 192 points and using a mean-zero function and a $\text{Matérn}_{\frac{3}{2}}$ covariance function. 4.06% of the observed values out of range.

The different scores against the number of KL coefficients retained are shown in Figure 4.20. In this case, all the scores follow the same decreasing tendency when increasing the number of KL coefficients used. The plots show that after around 18 KL coefficients, the scores are not changing significantly. Therefore, it seems sensible to build our training set based on 18-dimensional design points, i.e., $D = 18$ in the emulation diagram 4.11.

Figure 4.21 shows the scatterplot between *observed SF* = *predicted SF* for this case. The relative error between 1000 observed and predicted surface fluxes according to the number of design points retained is showed in Figure 4.22

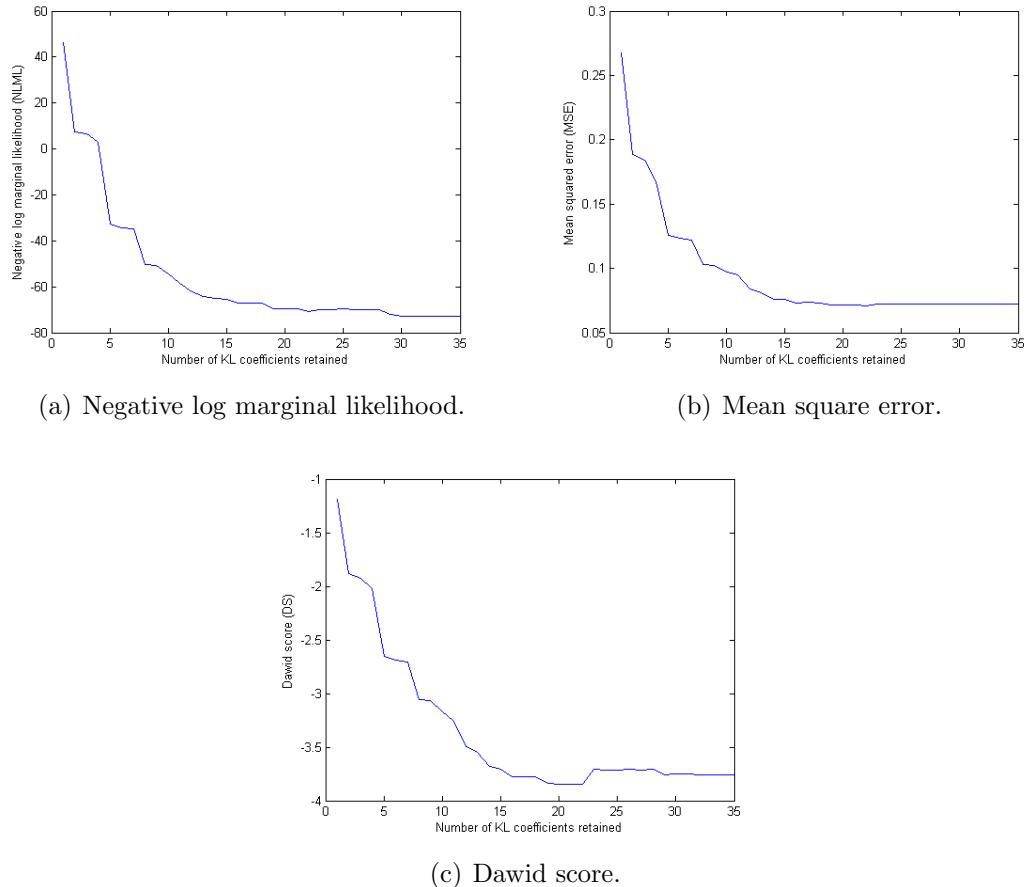


Figure 4.20: Different scores for a GP emulator built with a mean-zero function, RQ covariance function and 192 design points. The parameters of the C-ED simulator are: $Ra = 100$, $Da = 0.1$, $\beta_L = \pi/2$, and $\beta_T = \beta_L/10$. The x-axis represents the number of KL coefficients retained for each score.

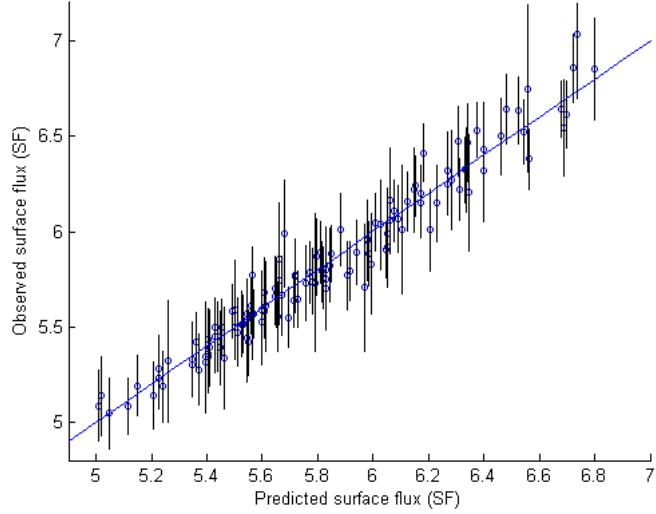


Figure 4.21: Observed and predicted surface fluxes for a design of 174 points, mean-zero function, and $\text{Matérn}_{\frac{3}{2}}$ covariance function. The solid line shows $\text{observed } SF = \text{predicted } SF$.

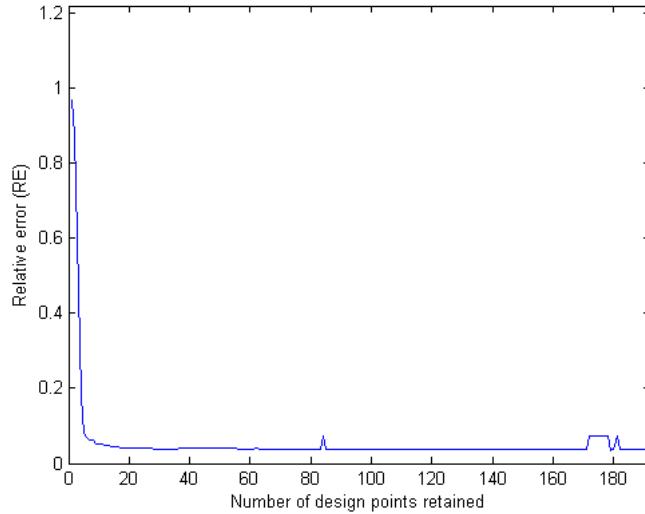


Figure 4.22: Relative error between 1000 observed and predicted surface fluxes against the number of design points retained using the RQ covariance function. The curve shows a decreasing tendency and it shows that after around 20 design points retained there is no variation in the relative error.

4.4.2 Uncertainty distribution of the *SF*. Case $Ra=100$

Figure (4.23) shows the MC UA for a sample of 1000 random permeability fields. The blue line is the estimation of the CDF of the *SF* and the dotted lines the 95% confidence interval for this empirical distribution.

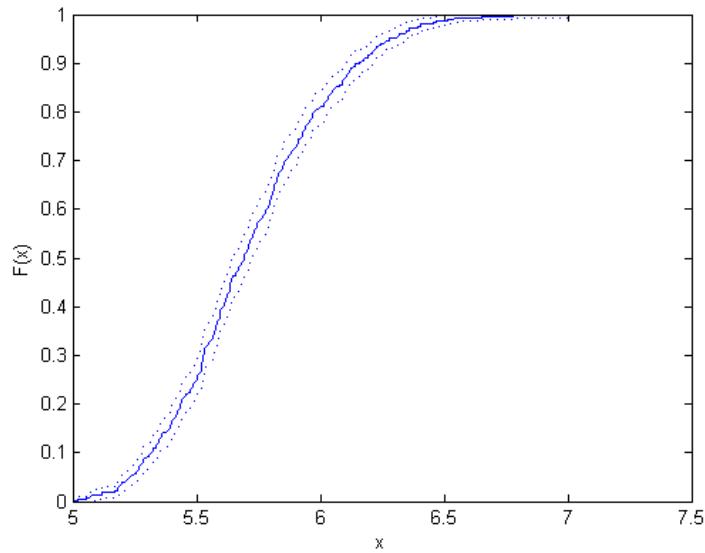


Figure 4.23: Monte Carlo ECDF (blue line) based on 1000 samples for the C-ED simulator. The dotted lines shows uncertainty bounds (95% confidence intervals).

In Figure 4.24 we show the difference between the results obtained by retaining 1 and 8 KL coefficients in the model. From this figure we see (graphically) how the more we increase the number of KL coefficients the more the uncertainty is reduced. Figures 4.24(a) and 4.24(c) show the ECDF (black) of the *SF* based on 1000 samples computed with the MC method, and GP posterior samples (green) for the GP emulator retaining 1 KL coefficient and 8 KL coefficients respectively. Figures 4.24(b) and 4.24(d) show the 2.5th and 97.5th percentiles (dashed magenta), the GP posterior mean (red) of the cumulative distribution function, and MC ECDF (black line) for 1 and 8 KL coefficients respectively.

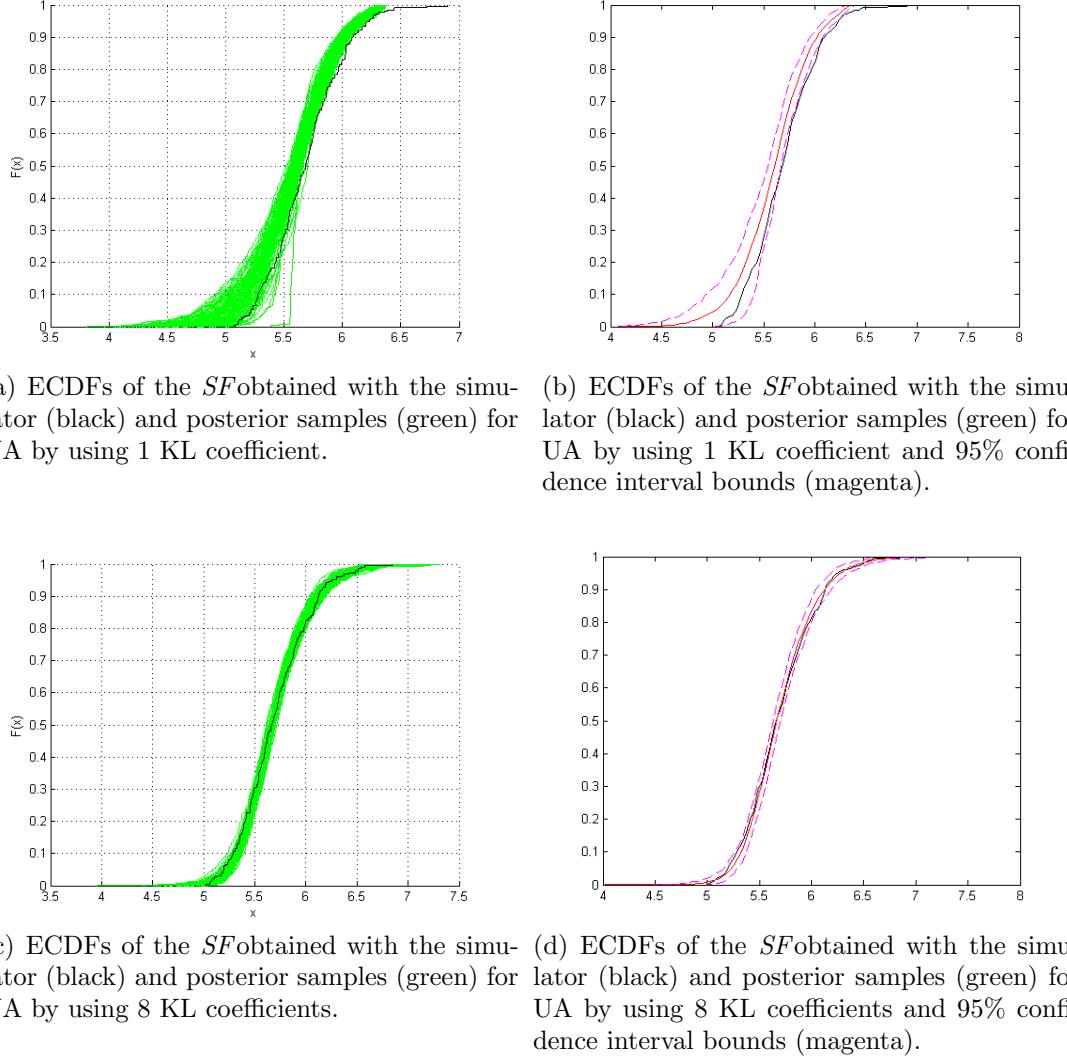


Figure 4.24: Uncertainty analysis for ECDF when considering 1 KL coefficients (a) and (b), and 8 KL coefficients (c) and (d).

To finish with the analysis of the C-ED problem, we will build a GP emulator for the whole problem without discarding any of the solutions. This is a big challenge for the GP emulator since it will have to predict the different scenarios (trivial or non-trivial solutions) for any given permeability. For being successful in this, GP regression is not sufficient, and we will have to complement GP regression with GP classification.

4.5 Gaussian process emulation for multiple solutions problems

The C-ED problem described with equations (4.10) and (4.11) is solved for the concentration, C , and the streamfunction, Ψ , as detailed in section 4.2.5. To be able to build a GP emulator without constraints on the different types of solutions, we need to label all possible outputs. By examining the solutions that we found numerically, discussed in Section 4.2.8, we clearly identify two different families of solutions, namely, solutions leading to a constant SF_0 and solutions leading to a $SF \neq SF_0$. Due to this singularity of the solutions we need to introduce an additional tool to our GP emulator, which we will call the *classifier*. The classifier will allow us to predict the class of the solution is likely to be, given the input. Once the solution is classified, we can then use the same approach followed in section 1.3.3 to estimate the value of the SF by using GP regression. In this section we will focus only in the case $Ra = 100$ (Note that the same methodology could be apply to $Ra = 60$). We will denote by SF_0 to the surface flux obtained from a no-flow solution and which value is 4.9707 in the case $Ra = 100$.

Let $\xi_M \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ a random vector formed by M KL coefficients, and let $\mathbf{K} \in \mathbb{R}^M$ be the corresponding RF generated from that ξ_M . Let $\mathcal{D} = [X, Y = f(X)]$ be the training set for our model. If we now split the set X in two disjoint sets, X_1 and X_2 , where $X_1 = \{\mathbf{x} \in X : f(\mathbf{x}) = SF_0\}$ and $X_2 = \{\mathbf{x} \in X : f(\mathbf{x}) \neq SF_0\}$, and we call $\mathcal{D}_2 = [X_2; Y_2]$, where $Y_2 = f(X_2)$, we can consider a GP (regression) emulator (as described so far), named GP_2 , based on the training set \mathcal{D}_2 . Thus,

for any given input ξ we run the classifier at first instance and if the classifier labels the output as a constant SF_0 we take the emulator output as $\hat{f}(\xi) = SF_0$, and if the output is labeled as a non-constant SF then we use GP_2 for making predictions, and we will take as predicted SF value, $\hat{f}_2(\xi) = SF$ (see Figure 4.25 for clarification).

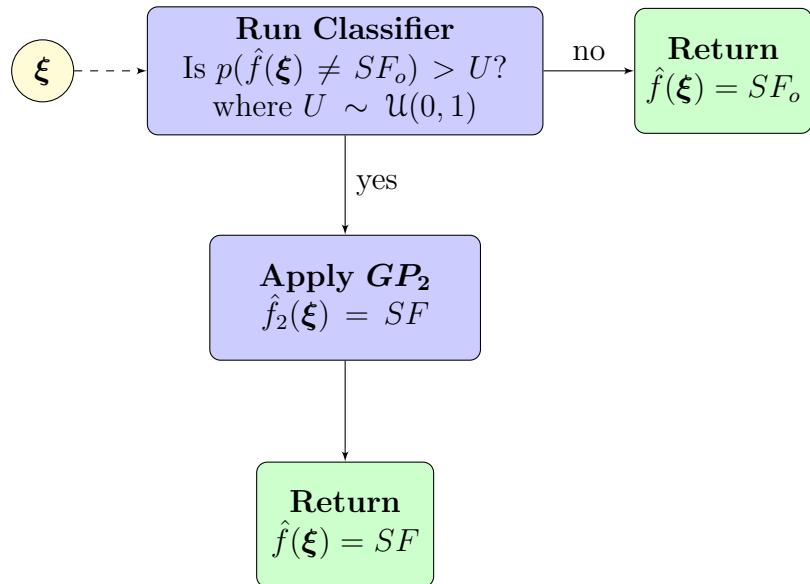


Figure 4.25: Algorithm followed for predicting the SF for any input given by using a Gaussian process classifier. $U \sim \mathcal{U}(0, 1)$ is an uniform random variable between 0 and 1 which gives randomness to the classification process.

4.5.1 Gaussian process classification

So far we have considered regression problems, where the targets were real valued. With the classification approach we wish to assign an input pattern \mathbf{x} to one of C classes, C_1, \dots, C_C . As mentioned earlier we are only interested at this time in two-class problem, $C_1 : SF = SF_o$ and $C_2 : SF \neq SF_o$. For creating a classifier we have followed the methodology described in Rasmussen and Williams (2006), chapter 3. We will use our GP classifier as probabilistic classifier, where test predictions probabilistic take the form of class probabilities. If we use the labels $y = +1$ and $y = -1$ to distinguish the two classes C_1 and C_2 , we will try to predict, for instance, $\pi(\mathbf{x}) = p(y = +1|\mathbf{x})$, the probability that an input \mathbf{x} is $y = +1$. As a GP prior over functions does not restrict the output to lie in the interval $[0,1]$, we need to "squash" the prior function¹ f . A common choice for this "squashing function" is the function $\lambda(z) = (1 + \exp(-z))^{-1}$, called the *logistic function*. So, the GP prior over f induces a prior over probabilistic classifications π . And then we can apply the methodology described in Section 1.3.3 to obtain the posterior mean for that $\pi(\mathbf{x})$. Thus, the difference between regression and classification is not of fundamental nature and actually we use a Gaussian process in essentially the same way, it is just that the Gaussian likelihood function often used for regression is inappropriate for classification. The likelihood function considered for our classification model will be the error-function² (or cumulative Gaussian), and does not take any hyperparameters.

Since exact inference is only possible for Gaussian likelihood we need an alternative approximation inference method. We will use the Expected Propagation (EP) algorithm (Minka, 2001) described in Rasmussen and Williams (2006), Section 3.6, for this purpose. The mean and covariance functions for the GP classification model are chosen to be a zero-mean function and the *SE* covariance function.

Let us see next some of the results after performing the GP emulator UA.

¹In practical terms, what we squash are the samples, f_i , drawn from the prior distribution of f .

²The error function, Erf , of a Gaussian distribution is defined as $Erf(x) = \frac{1}{\sqrt{\pi}} \int_{-x}^x e^{-t^2} dt$.

4.5.2 UA of the ECDF of the SF for the C-ED problem

The final results after performing a full UA of the CDF of the SF for the C-ED model with parameters $Ra=100$, $Da=0.1$, $\beta_L = \pi/2$ and $\beta_T = \beta_L/10$ are showed next. Figure 4.26(a) illustrates how GP regression is unable to predict the true distribution around the bifurcation point. Figure 4.26(b) shows how posterior samples from the improved method are now able to predict the bifurcation around $SF_0 = 4.9707$. Finally, Figure 4.27 shows the 2.5th and 97.5th percentiles (dashed), the median of the predicted distribution (red), and MC ECDF (black line).

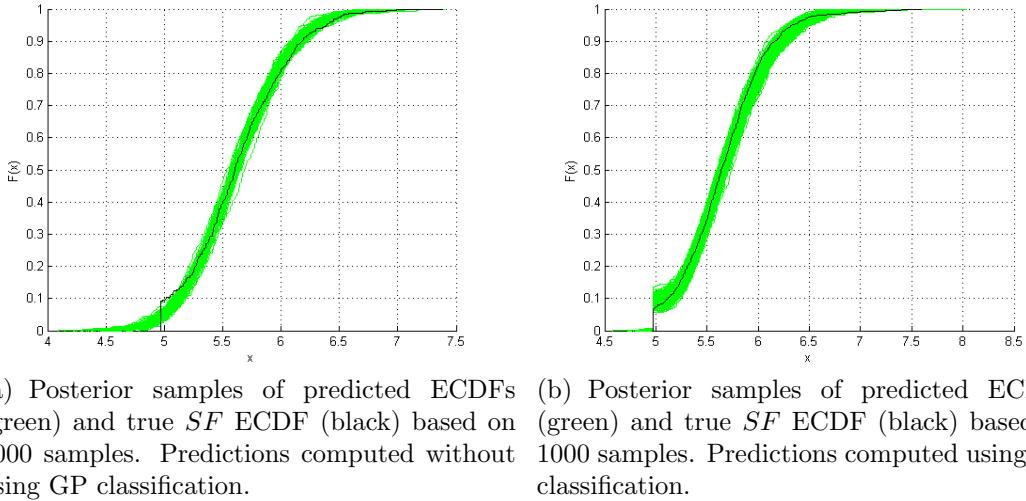


Figure 4.26: Difference between posterior samples of predicted ECDFs (green) and true SF ECDF (black) based on 1000 samples with and without using GP classification. The number of design points were 256 and the number of KL coefficients were 18. The parameters for the E-CD problem were chosen to be $Ra=100$, $Da=0.1$, $\beta_L = \pi/2$ and $\beta_T = \beta_L/10$. Priors mean-zero, SE covariance, Erf likelihood functions and EP method for inference were chosen for the GP classifier. Priors mean-zero, RQ covariance, Gaussian likelihood functions and exact inference method were chosen for GP regression model.

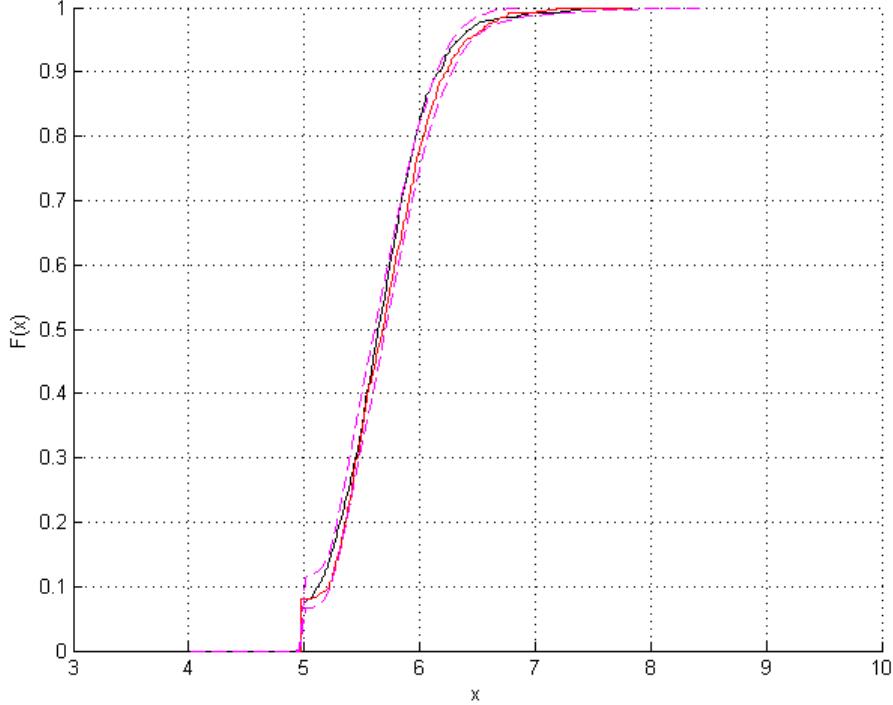


Figure 4.27: UA of predicted SF ECDFs based on 1000 samples using GP classification. True ECDF (black), predicted ECDF (red), 2.5th and 97.5th percentiles (dashed magenta). The number of design points were 256 and the number of KL coefficients were 18. The parameters for the E-CD problem were chosen to be $Ra=100$, $Da=0.1$, $\beta_L = \pi/2$ and $\beta_T = \beta_L/10$. Priors mean-zero, SE covariance, Erf likelihood functions and EP method for inference were chosen for the GP classifier. Priors mean-zero, RQ covariance, Gaussian likelihood functions and exact inference method were chosen for GP regression model.

The last analysis of this thesis will be related to the comparison between the direct scalar emulation of the SF and the estimation of the SF from emulated concentration fields with the methodology described and used in Chapter 2 for the travel time and the pressure field.

4.6 Gaussian process emulation of concentration and streamfunction fields

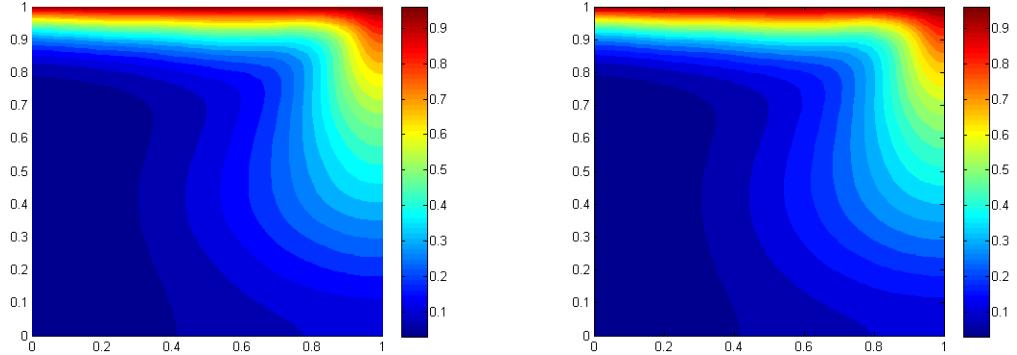
Let $f(\cdot)$ be the concentration (or streamfunction) field simulator, i.e., the simulator which takes inputs \mathbf{x} and returns an output \mathbf{y} , where \mathbf{y} is now (either) a 2-dimensional concentration (or streamfunction) field solution of our C-ED problem. If we use the SVD method described in Section 2.11.1 to build an emulator from \mathbf{x} to \mathbf{y} , we can then compute the *SF* by using formula (4.19).

In the following, we show some examples of the reduced rank approximations obtained when applying the SVD method to a set of observed concentration fields, as well as the *RE* achieved in the approximations.

We consider two of the concentration fields, C_1 and C_2 , obtained by solving equations (4.10) and (4.11) with parameters: $Ra = 100$, $Da = 0.1$, $\beta_L = \frac{\pi}{2}$ and $\beta_T = \frac{\beta_L}{10}$. Figures 4.28(a) and 4.29(a) show the true concentration fields, and Figures 4.28(b) and 4.29(b) the corresponding reduced rank approximations, \tilde{C}_1 and \tilde{C}_2 . The *RE* for C_1 and \tilde{C}_1 is 0.0014 and the *RE* for C_2 and \tilde{C}_2 is 0.0013.

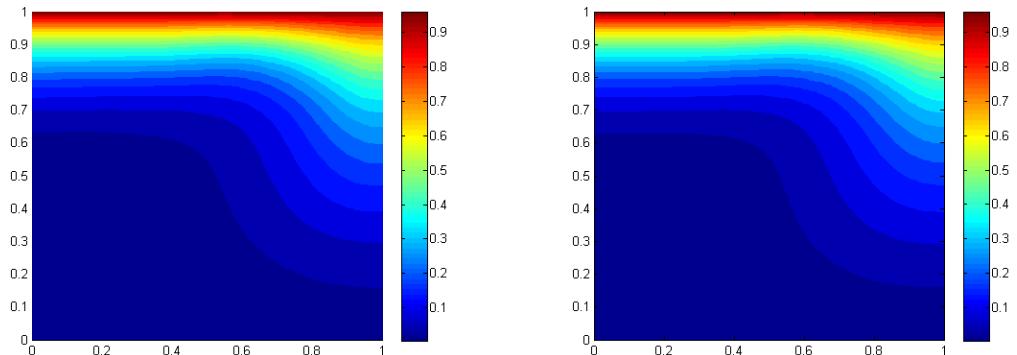
We also show the results obtained when applying the method to the streamfunction fields. Figures 4.30(a) and 4.31(a) are the true streamfunction fields, P_1 and P_2 , and Figures 4.30(b) and 4.33(b) the corresponding reduced rank approximations, \tilde{P}_1 and \tilde{P}_2 . The *RE* for P_1 and \tilde{P}_1 is 0.0097 and the *RE* for P_2 and \tilde{P}_2 is 0.0037.

In the following section, we build the GP field emulator for the CE-D model and we show some illustrative results.



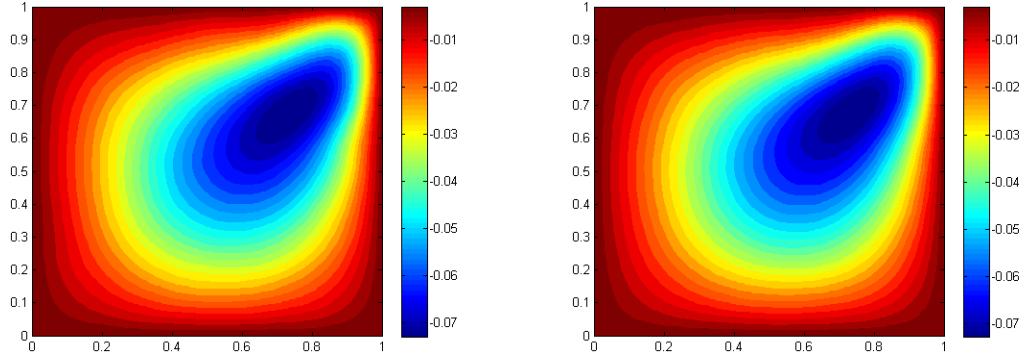
(a) True concentration field, C_1 , obtained from the simulator for the E-CD problem for $Ra=100$, $Da=0.1$, $\beta_L = \pi/2$ and $\beta_T = \beta_L/10$.
(b) Reduced rank approximation, \tilde{C}_1 , of the concentration field C_1 by using 15 eigenvectors.

Figure 4.28: True concentration field, C_1 , and corresponding reduced rank approximation, \tilde{C}_1 . Number of eigenvectors used is $k = 15$. The L^2 -norm relative error between C_1 and \tilde{C}_1 is 0.0014.



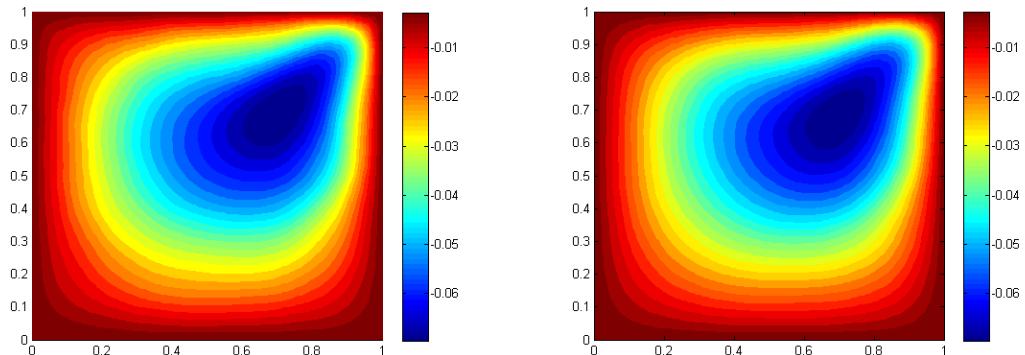
(a) True concentration field, C_2 , obtained from the simulator for the E-CD problem for $Ra=100$, $Da=0.1$, $\beta_L = \pi/2$ and $\beta_T = \beta_L/10$.
(b) Reduced rank approximation, \tilde{C}_2 , of the concentration field C_2 by using 15 eigenvectors.

Figure 4.29: True concentration field, C_2 , and corresponding reduced rank approximation, \tilde{C}_2 . Number of eigenvectors used is $k = 15$. The L^2 -norm relative error between C_2 and \tilde{C}_2 is 0.0013.



(a) True concentration field, P_1 , obtained from the simulator for the E-CD problem for $Ra=100$, $Da=0.1$, $\beta_L = \pi/2$ and $\beta_T = \beta_L/10$.
(b) Reduced rank approximation, \tilde{P}_1 , of the concentration field P_1 by using 15 eigenvectors.

Figure 4.30: True concentration field, P_1 , and corresponding reduced rank approximation, \tilde{P}_1 . Number of eigenvectors used is $k = 15$. The L^2 -norm relative error between P_1 and \tilde{P}_1 is 0.0097.



(a) True concentration field, P_2 , obtained from the simulator for the E-CD problem for $Ra=100$, $Da=0.1$, $\beta_L = \pi/2$ and $\beta_T = \beta_L/10$.
(b) Reduced rank approximation, \tilde{P}_2 , of the concentration field P_2 by using 15 eigenvectors.

Figure 4.31: True concentration field, P_2 , and corresponding reduced rank approximation, \tilde{P}_2 . Number of eigenvectors used is $k = 15$. The L^2 -norm relative error between P_2 and \tilde{P}_2 is 0.0037.

4.7 Building the Gaussian process field emulator for the C-ED problem

Let f be the previous concentration field simulator, which takes an input \mathbf{x} , and returns an output \mathbf{y} , where \mathbf{y} is a concentration field. Suppose we have a reduced rank approximation of n concentration fields, $\mathbf{y}_1, \dots, \mathbf{y}_n$, given by the following expression:

$$\tilde{\mathbf{y}}_j \approx L_* D_* \mathbf{t}_j, \quad \text{for } i = 1, \dots, n, \quad (4.23)$$

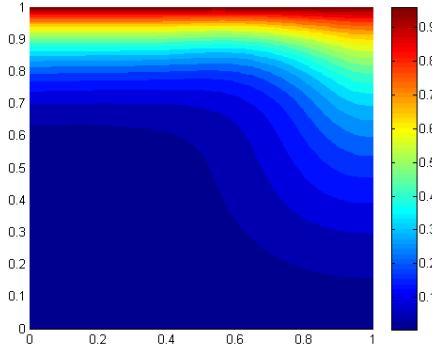
where $L_* = (\mathbf{l}_1, \dots, \mathbf{l}_n)$ and $D_* = (\mathbf{d}_1, \dots, \mathbf{d}_k)$, for some $k < n$, and \mathbf{t}_j is a vector of length k (as discussed in Section 2.11.1).

To build an emulator, f^* , for the simulator, we can build an emulator by using \mathbf{x} and the n vectors \mathbf{t}_j as a training set. Then, we build k separate emulators, f_j^* , for $j = 1, \dots, k$, with their respective training sets formed by \mathbf{x} and each of the elements of the rows of the matrix $R_*^\top = (\mathbf{t}_1, \dots, \mathbf{t}_n)$. For a new given input \mathbf{x}^* we can compute $f_j^*(\mathbf{x}^*) := t_j^*$, for $j = 1, \dots, k$, and form $f^*(\mathbf{x}^*) := (f_1^*(\mathbf{x}^*), \dots, f_k^*(\mathbf{x}^*)) = \mathbf{t}^*$, where $\mathbf{t}^* := (t_1^*, \dots, t_k^*)$. So, given \mathbf{t}^* , the estimated centered field will be given by,

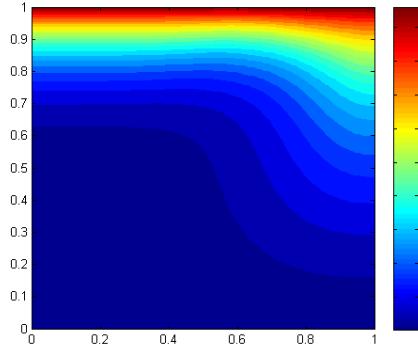
$$\tilde{y}^* = f^*(\mathbf{x}^*) = L_* D_* \mathbf{x}^*. \quad (4.24)$$

Figures 4.32 and 4.33 show some examples containing the true, reduced rank and predicted concentration and streamfunction fields. Figures 4.34 shows how the GP field emulator is even able to predict the existence of two cells streamfunctions.

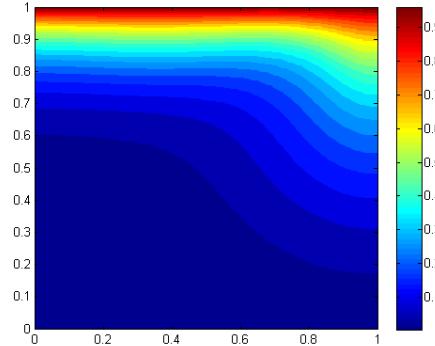
We finish this chapter by presenting the results obtained when using the two proposed methods for predicting the SF value for our C-ED problem, GP scalar and field emulation.



(a) Observed concentration field, C , obtained from the simulator for the E-CD problem for $Ra=100$, $Da=0.1$, $\beta_L = \pi/2$ and $\beta_T = \beta_L/10$.

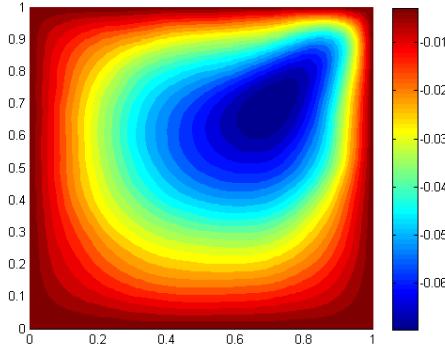


(b) Reduced rank approximation, \tilde{C} , of the observed concentration field C by using 40 eigenvectors.

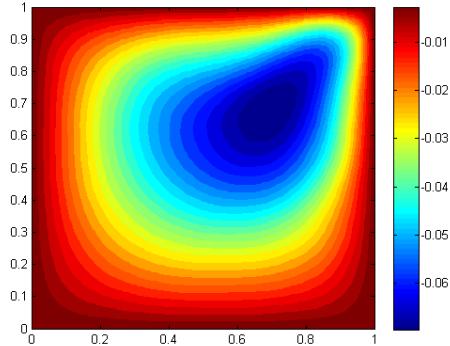


(c) Predicted concentration field, C^* , of the concentration field C by using 40 eigenvectors and 16 KL coefficients.

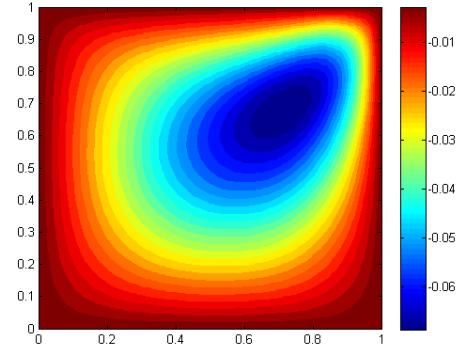
Figure 4.32: Observed concentration field, C , and corresponding reduced rank approximation, \tilde{C} , and predicted concentration field, C^* . Number of eigenvectors used is $k = 40$. Number of KL coefficients for prediction is 16. The L^2 -norm relative error between C and \tilde{C} is 0.0031 and the L^2 -norm relative error between C and C^* is 0.0370.



(a) Observed streamfunction field, Ψ , obtained from the simulator for the E-CD problem for $Ra=100$, $Da=0.1$, $\beta_L = \pi/2$ and $\beta_T = \beta_L/10$.

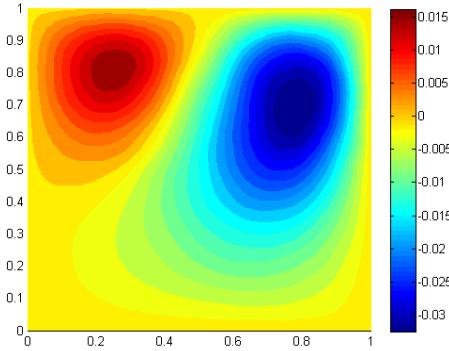


(b) Reduced rank approximation, $\tilde{\Psi}$, of the observed concentration field Ψ by using 40 eigenvectors.

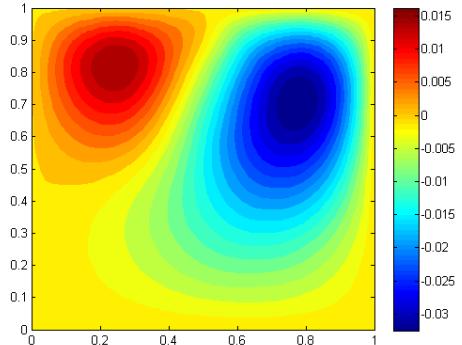


(c) Predicted streamfunction field, Ψ^* , of the concentration field Ψ by using 40 eigenvectors and 16 KL coefficients.

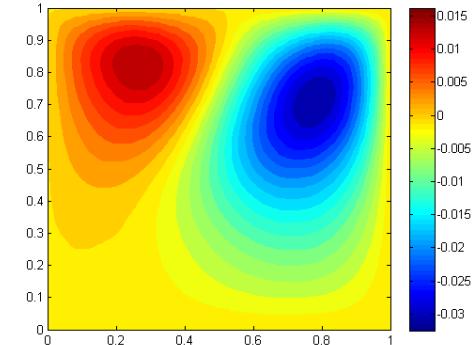
Figure 4.33: Observed streamfunction field, Ψ , and corresponding reduced rank approximation, $\tilde{\Psi}$, and predicted streamfunction field, Ψ^* . Number of eigenvectors used is $k = 40$. Number of KL coefficients for prediction is 16. The L^2 -norm relative error between Ψ and $\tilde{\Psi}$ is 0.0111 and the L^2 -norm relative error between Ψ and Ψ^* is 0.1887.



(a) Observed two cells streamfunction field obtained from the simulator for the E-CD problem for $Ra=100$, $Da=0.1$, $\beta_L = \pi/2$ and $\beta_T = \beta_L/10$.



(b) Reduced rank approximation by using 40 eigenvectors.



(c) Predicted streamfunction field by using 40 eigenvectors and 16 KL coefficients.

Figure 4.34: Observed, reduced rank approximation and predicted streamfunction fields. Number of eigenvectors used is $k = 40$. Number of KL coefficients for prediction is 16. The L^2 -norm relative error between the observed field and reduced rank approximation is 0.0031 and the L^2 -norm relative error between the observed field and the predicted is 0.1521.

4.8 Comparison between directly predicted surface flux and estimated surface flux from emulated concentration field

For this comparison we used 500 *SF* outputs of the C-ED simulator with parameters: $Ra=100$, $Da=0.1$, $\beta_L = \pi/2$ and $\beta_T = \beta_L/10$. The training set chosen for the Gaussian processes was formed by 123 concentration fields¹. The prior specifications for both Gaussian processes were: zero-mean, rational quadratic covariance and Gaussian likelihood functions. The number of KL coefficients retained was 16. And finally, the number of eigenvectors used for the SVD rank reduction was 40.

With these specifications the *RE* for the scalar GP was 0.0182 meanwhile the *RE* for the field GP was 0.0201. Thus, the conclusion for this experiment is that scalar GP emulation seems to be more efficient than field GP emulation, although both of them seem to be a powerful tool for making predictions. Figures (4.36(a) and 4.36(b)) show visual comparison of both results.

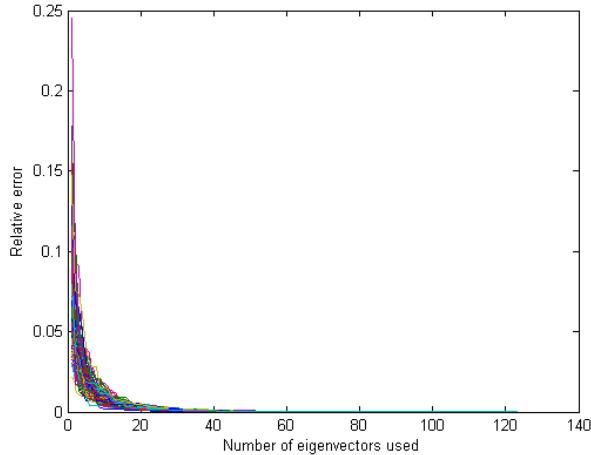


Figure 4.35: *RE* (Y axis) between each of the observed concentration field and its corresponding reduced rank approximation along the number of eigenvectors used (X axis).

¹This number of 123 fields correspond to the remaining number of concentration fields, from the original set of 256 design points, that did not lead to a constant *SF*.

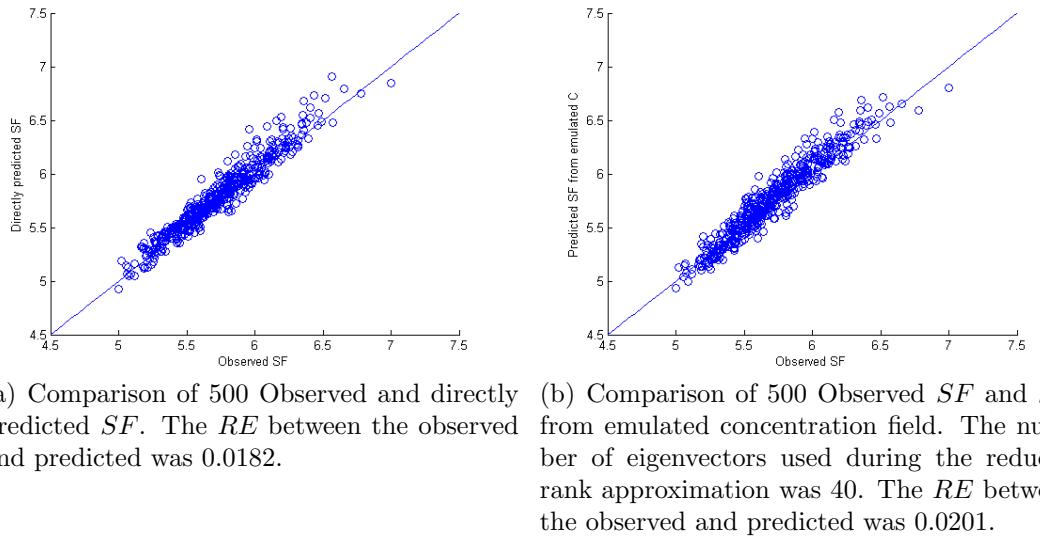


Figure 4.36: Observed SF vs directly predicted SF and Observed SF vs SF from predicted concentration field plots obtained from the simulator for the E-CD problem for $Ra=100$, $Da=0.1$, $\beta_L = \pi/2$ and $\beta_T = \beta_L/10$. The design was formed by 123 points (one cell solutions only). A zero-mean and RQ covariance function were used as prior GP model specifications and the number of KL coefficients used for prediction was 16. The solid line shows $observed\ SF = predicted\ SF$.

Chapter 5

Conclusions and further work

The contribution of this thesis can be classified in three main blocks: first, the search and implementation of computational tools for performing uncertainty quantification in groundwater flow problems. In particular, this thesis has analysed and compared the efficiency of four methods, namely, MC, MLMC, QMC and MLQMC for computing the average travel time of a convected particle in groundwater flow. Second, the study and analysis of a more computationally complex real problem, the C-ED process occurring in CO₂ geological storage. Here, we have been more interested in understanding the physical meaning and behaviour of the solutions, and how these solutions change for different inputs. And third, the development of an alternative method to the four above for doing uncertainty quantification, the GP emulator. In this thesis we aimed to quantify how the uncertainty (or heterogeneity when studied as an individual case) in the permeability field, modelled as a log Gaussian random field, affect to either the travel time, in the groundwater flow model, or the surface flux in the C-ED process, when it is propagated through the model equations. In the following, we will discuss each of these three blocks in more detail and give indication of further lines of research.

5.1 Computational techniques for UQ

In Chapter 3 we used four approaches, namely, MC, MLMC, QMC and MLQMC to approximate the average travel time that a convected particle released at the centre of a given physical domain lasts to get the domain boundary. A series of experiments based on several numerical discretisations of the physical domain were conducted and the four above approaches compared. Numerical results demonstrated that a number of improvements to the standard MC simulation are possible, these include, for instance, the QMC method, where the parameter values are chosen in a deterministic way to speed up the rate of convergence of the MC method. Another improvement, the MLMC method, based on the linearity of the expectation and inspired by the multi-grid approach in numerical PDEs, exploits the structure of the PDEs in order to reduce the amount of computational work required. MLMC can be seen as a variance reduction technique for the standard Monte Carlo method (Cliffe et al., 2011). The accuracy of the approximations and the efficiency of the four methods were expressed in terms of the ε^2 -cost for different predefined tolerances, specified through the MSE.

Numerical results showed that all the proposed alternatives to the standard MC simulation, namely, QMC, MLMC and MLQMC, applied to the groundwater flow model described in Chapter 2, improved significantly the standard MC simulation, providing the same order of accuracy with a lower computational cost. The overall picture with the performance of all the methods was shown in Figure 3.6. The plot showed how the MLQMC method produces the lowest computational cost for all the tolerances. MLMC was found to perform better than MC and QMC, and in conclusion, MC was proven to be the least efficient method for this model. However, both multilevel based methods, MLMC and MLQMC, require the complete rewriting of the simulator's numerical code in order to carry out the corresponding multi-grid approaches, this may be a handicap for end-users working in commercial environments where the use of engineering commercial packages, sometimes used as 'black boxes', makes the use of this accelerators unfeasible. Another historical constraint of multilevel methods has been¹ that

¹A recent publication (Giles et al., 2015) describes a technique to use MLMC to approximate distribution functions and densities of solutions of PDEs.

these methods do not produce the density or cumulative distribution functions of the uncertain output and therefore they only can be used to approximate the expected value of a functional of the PDEs solution. On the other hand, MC and QMC can be used to find out the entire distribution function of the quantity of interest although this sometimes for complex simulator would be impractical. This latter strengthen the use of computationally cheap surrogates instead, like GP emulators, for complex simulators as the one proposed in Chapter 4 to model the C-ED process.

Following above considerations, further research can be done by trying to apply the recent technique described in Giles et al. (2015) to the models described in this thesis, and do the comparisons done in Sections 2.10.2 and 4.5.2 with MLMC, and also extend this technique to the MLQMC method.

Due to the deterministic way in which the QMC and MLQMC methods are designed, it seems sensible to think that it is possible to improve these methods in order to minimize (or control) the number of samples to run at each level for reducing the statistical error, instead of using just an exceeding estimation provided by expression (1.12). It is also interesting to try with an alternative sampling technique to Sobol sequences when building QMC estimator, and by analogy, the GP emulator design which is also built upon Sobol sequences.

5.2 Analysis of the C-ED process in CO₂ geological storage

In Chapter 4, we searched for numerical solutions of the C-ED problem in terms of the streamfunction and concentration for different permeability fields and investigated how spacial variations (or heterogeneity) of the permeability fields affected the *SF*. The existence of multiple solutions added additional challenges to the search of numerical solutions others than the trivial solution (called the no-flow solution). The FEM code was only able to find one solution (the no-flow) for the problem, and thus, to overcome this, it was necessary to use other techniques

in conjunction with the FEM, the arclength continuation methodology as a tool for finding numerical solutions others than the no-flow solution. In Figure 4.2 we showed a representation of all the solutions found. This technique allowed us to discover the existence of many branches of solutions dependent of one parameter, in this case, the Ra . We analysed some of the possible solution scenarios arisen for different heterogeneous input permeability fields and measured the impact that the heterogeneity had on the SF by comparing the results with the homogeneous case, i.e., considering the permeability constant in all the domain. The value of that constant was chosen to be the average of the values representing the heterogeneous permeability field to compare.

Numerical results presented in Chapter 4 showed that the magnitude of the local values in the permeability field is not as important as the existence of interconnected regions with similar permeability value. These interconnected regions which cross the problem domain from one edge to the other, even if they are either narrow or regions with low permeability values, provoke directly the changes on the magnitude of the resulting surface flux.

Further work can be done along this line of research by investigating if these bifurcation scenarios found for low values of Ra ($Ra=60$ and $Ra=100$) are repeated again for larger values of Ra , i.e., attempting to reproduce the bifurcation diagrams presented in Ward et al. (2014).

5.3 Gaussian process emulation for UQ

One of the goals of this thesis has been to quantify the uncertainty on the CO_2 dissolution flux that goes into the brine during CO_2 geological storage induced by the uncertainty on the permeability fields. For that purpose, a finite element computer simulator of two-dimensional convection in a finite-depth porous medium was developed and described in Chapter 4. The uncertain input permeability field for this simulator was modelled as a log Gaussian random field. We examined the C-ED process by looking at the expected distribution of the dissolution flux. In this model the uncertainty in the permeability is propagated to

the solution of the system of PDEs modelling the process, and thus makes us uncertain about the expected dissolution flux. The proven impracticability (it took us approximately 14 full days to run 1000 times the C-ED simulator described in Chapter 4) of the MC method for computing the expected distribution of the dissolution flux led us to search for an alternative methodology, namely, Gaussian process emulation. This alternative method for doing UQ, based around the use of emulators (sometimes known as meta models), was proven to be a cheap and efficient statistical surrogate for the CE-D simulator model. The emulator offered the possibility of being run as many times as necessary in order to perform any analysis we were interested in doing with the simulator, for instance UQ.

Before building the GP emulator for the C-ED model, we showed how to build a GP emulator for a simpler model, the groundwater flow model, and demonstrated that accurate UQ can be carried out at considerably reduced cost compared to the Monte Carlo analysis. To build the emulator we used Gaussian process regression methodologies consisting in establishing a prior specification of the functional form of the model which was updated in the light of data provided (the observed values). This prior specification for the GP emulator consists in providing the model with a mean and covariance structure and a set of observed values for some given inputs. In this thesis we have used a mean-zero function and four different covariance structures, namely, SE , $Matérn_{\frac{3}{2}}$, $Matérn_{\frac{5}{2}}$, and RQ . To test the performance of our GP emulator as statistical surrogate of the groundwater flow model simulator, we used the GP emulator to estimate the average travel time of a convected particle in the same way we did with MC, this time reporting the emulator's predictions in terms of 95% CI. These results were showed in Figure 3.7 where a general picture of the convergence of the MC method for various tolerances was given, as well as the corresponding estimations of the average travel time for the four prior specifications discussed above. All the predictions given by the emulator were within the RMSE bars, and therefore we concluded that the GP performance was satisfactory.

Once the emulation technique was satisfactorily applied to the groundwater flow model described in Chapter 2, we built the same GP emulator for the C-ED process. Figure 4.26(a) showed that an emulator built only upon GP regression was not enough to produce satisfactory results as it was unable to predict

the model bifurcation adequately. Thus, it was necessary to extend the previous methodology to the use of GP classification in addition to GP regression. Figure 4.26(b) shows the improvement and how the new emulator, built now by using jointly GP regression and classification, was able to predict the uncertainty around the bifurcation.

Further work can be done in terms of reducing the uncertainty provided by the GP emulator presented in this thesis by trying with different prior model, likelihood function or inference methods. These alternatives are gathered in Rasmussen and Williams (2006).

One of the main challenges of using GP emulators found during this thesis was to estimate correctly the hyperparameters. For high dimensional hyperparameters (around 20 KL coefficients retained) the GPML optimizer routine used to estimate the value of the hyperparameters did not perform well and made the GP emulator to give bad estimations of the quantity of interest. In this thesis, it was not necessary to employ more than 18 KL coefficients to build a satisfactory GP emulator but it is convenient to have this fact in mind when dealing with other models or quantities of interest that may require further refinement of the model parameters.

It is also worth to mention that our GP emulator once it had reached a certain level of accuracy with respect to the observed values it did not improve performance when increasing the number of design points used, i.e., the number of observed values considered in the training set. Normally, the performance kept improving along with the number of design points used in the training set until a certain number, normally, between 30 and 50 design points, after this number, the model did not improve performance in terms of the MSE when compared with a set of observed values (see Figure 2.12). This could indicate that there is not much variation on the observed values for the inputs given, and that a small number of points is enough to infer the pattern of the rest of the points for this model. Said that, it could be useful for other models, which may need a higher number of design points, to use alternative schemes for building the training set, for instance, the Latin Hypercube Sampling (McKay et al., 1979) described by Pebesma and Heuvelink (1999).

Other technique that we did not have time to investigate in this thesis, and

may reduce considerably the computational cost of the GP emulators, is the extension of the concept of multilevel techniques to the GP emulator, i.e., attempt to build a multilevel GP emulator for the models presented in this thesis. This approaches were discussed in Cumming and Goldstein (2009) for instance.

A novel methodology, based upon the use of GP emulation and singular value decomposition algorithms, for predicting field outputs (like pressure, concentration or streamfunction fields for instance) instead of scalar outputs (like travel time or surface flux) for a given model was also introduced in Chapters 2 and 4. An experiment was conducted to investigate which of the two methods, direct scalar emulation or field emulation, was more efficient. The experiment consisted in computing the quantity of interest with both approaches and compare the resulting quantities with those given by the simulator. Numerical calculations showed that direct scalar emulation produce a lower MSE when compared with true observed solutions. Besides this method was proven to be less efficient than direct scalar emulation in terms of computing average values of the solutions, it offers an invaluable tool to recover, through the prediction of the output field for a given input, the physical meaning of the process which direct emulation cannot provide. Along this latter, a possible and challenging line of future research arises on the investigation of the extension of this methodology to three dimensional computationally complex models.

Abbreviations

<i>ARD</i>	Automatic relevance determination
<i>C-ED</i>	convectively-enhanced dissolution
<i>CDF</i>	cumulative distribution function
<i>CI</i>	confidence interval
<i>CCS</i>	Carbon capture and storage
CO_2	Carbon dioxide
<i>CV</i>	cross validation
<i>DS</i>	Dawid score
<i>ECDF</i>	empirical cumulative distribution function
<i>FE</i>	finite element
<i>FEM</i>	finite element method
<i>FV</i>	finite volume
<i>GP</i>	Gaussian process
<i>GPML</i>	Gaussian process for machine learning (code)

<i>KL</i>	Karhunen-Loéve
<i>LHS</i>	Latin hypercube sampling
<i>LOO-CV</i>	leave one out-cross validation
<i>MSE</i>	mean squared error
<i>MC</i>	Monte Carlo
<i>MLMC</i>	multilevel Monte Carlo
<i>QMC</i>	quasi Monte Carlo
<i>MLQMC</i>	multilevel quasi Monte Carlo
<i>NLML</i>	negative log marginal likelihood
<i>PCA</i>	principal components analysis
<i>PDE</i>	partial differential equation
<i>PDF</i>	probability density function
<i>RF</i>	random field
<i>RQ</i>	rational quadratic
<i>RMSE</i>	root mean squared error
<i>SE</i>	squared exponential
<i>SVD</i>	singular value decomposition
<i>UA</i>	uncertainty analysis
<i>UQ</i>	uncertainty quantification

Symbols and notation

Matrices are capitalized and vectors are in bold type. A subscript asterisk, such as in \mathbf{x}_* or X_* , indicates reference to a test set quantity.

Symbol	Meaning
$ A $	determinant of matrix A
$\ \mathbf{x}\ _2$	Euclidean norm of vector \mathbf{x} , i.e. $(\sum_i \mathbf{x}_i^2)^{1/2}$
$\ \mathbf{x} - \mathbf{x}'\ _2$	Euclidean distance between two points \mathbf{x} and \mathbf{x}' , i.e. $(\sum_i (\mathbf{x}_i - \mathbf{x}'_i)^2)^{1/2}$
\mathbf{x}^\top	the transpose of vector \mathbf{x}
\sim	distributed according to; example $x \sim \mathcal{N}(m, \sigma^2)$
∇	gradient or partial derivatives
α	order of convergence for the MC method
α_s	value for the $100\alpha_s$ percentile, p_{α_s} , s.t. $F(p_{\alpha_s}) = \alpha_s$
b	thickness of a layer of rock
β	order of convergence of the variance of Y_ℓ
β_{tol}	tolerance for KL decomposition truncation error
β_c	volumetric expansion coefficient
β_T	transverse dispersion length
β_L	longitudinal dispersion length
c	covariance function in KL decomposition
C	concentration
\mathbf{C}	covariance matrix in KL decomposition
C_{ij}	components of the covariance matrix in KL decomposi- tion

\cosh	hyperbolic cosine
cov	covariance of the noisy observations \mathbf{y}
$cov(\mathbf{f}_*)$	Gaussian process posterior covariance
\mathcal{C}_ε	computational ε -cost for a RMSE of $e(\hat{\mathbf{T}}_M) \leq \varepsilon$
\mathcal{C}_ℓ	cost of a single sample of Y_ℓ
$\mathcal{C}(\mathbf{T}_M^{(i)})$	cost to compute one sample $\mathbf{T}_M^{(i)}$
$\mathcal{C}(\hat{\mathbf{T}}_{M,N}^{MC})$	cost of MC estimator
C_0	concentration at top boundary and characteristic unit for C
C_j	concentration field j
\tilde{C}_j	reduced rank approximation of concentration field C_j
C^*	emulated concentration field
d	number of training cases or design points
d_*	number of test cases
D	dimension of input space (or dimension of design points). Also referred to the model domain in Chapters 2 and 4. Also referred to the matrix of singular values in the SVD method
D_*	reduced rank approximation od matrix D
Da	Damköler number
\mathbf{D}	dispersion tensor
D_0	characteristic unit for \mathbf{D}
\mathcal{D}	training set
\mathfrak{D}	matrix form of the training set
δ_{ij}	Kronecker delta, $\delta_{ij} = 1$ iff $i = j$ and 0 otherwise
E	relative error in the truncation of the KL decomposition
\mathbf{e}_z	unitary vector (ordinate axis)
\mathbb{E}	expectation
$e(\hat{\mathbf{T}}_M)$	RMSE of $\hat{\mathbf{T}}_M$
\mathcal{F}	set of events (σ -algebra)
$f(\cdot)$	simulator
\hat{f}	emulator mean posterior
$f(\mathbf{x})$	GP latent function values
$f(\mathbf{x}) _D$	general GP posterior, prior updated in the light of D
\mathbf{f}	Gaussian vector of latent function values, $\mathbf{f} = (f(\mathbf{x}_1), \dots, f(\mathbf{x}_n))^\top$
\mathbf{f}_*	Gaussian posterior distribution of the predictions at test locations

$\mathbf{f}_* \mathbf{X}, \mathbf{y}, \mathbf{X}_*$	conditional posterior distribution for test inputs \mathbf{X}_*
$f_{(i)}$	random functions (samples) from posterior distribution
\bar{f}_*	Gaussian posterior prediction for a single test point
$\bar{\mathbf{f}}_*$	Gaussian posterior distribution mean
F	CDF of a random variable
\hat{F}	ECDF of a random sample
G	probability distribution of the random input variable
γ	order of convergence of one sample $T_M^{(i)}$
γ_c	reaction rate
\mathbf{g}	acceleration due to gravity
\mathbf{g}_s	source terms
\mathcal{GP}	Gaussian process: $f \sim \mathcal{GP}(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}'))$, the function f is distributed as a Gaussian process with mean function $m(x)$ and covariance function $k(\mathbf{x}, \mathbf{x}')$
h	hydraulic head
H	depth of the model domain D in Chapter 2 and half of depth of the domain D in Chapter 4
H_M	number of terms after truncating the KL decomposition
\mathbb{I}	Indicator function or unit (identity) tensor
\mathbf{J}	Fickian mass flux
\mathbf{J}_x	first component of \mathbf{J}
\mathbf{J}_z	second component of \mathbf{J}
K	parameter permeability
K_0	characteristic unit for K
\mathbf{K}	discrete log-Gaussian permeability field
$\bar{\mathbf{K}}_j$	homogeneous permeability field with constant values equal to the averaged value of the discrete log-Gaussian permeability field \mathbf{K}_j
$k(\mathbf{x}, \mathbf{x}')$	prior covariance function evaluated at \mathbf{x} and \mathbf{x}'
k_D	the posterior covariance function of the Gaussian process
k_{SE}	squared exponential covariance function
$k_{\frac{3}{2}}$	Matérn class 3/2 covariance function
$k_{\frac{5}{2}}$	Matérn class 5/2 covariance function
k_{RQ}	rational quadratic covariance function
L	length of the domain Ω
L	left singular matrix in the SVD method
L_*	reduced rank approximation od matrix L

L_F	number of realisations for obtaining any inference about $F(\cdot)$
\mathcal{L}	aspect ratio
\log	natural logarithm (base e)
ℓ	characteristic length scale
$\boldsymbol{\ell}$	d -dimensional characteristic length scale
ℓ_i	i^{th} component of $\boldsymbol{\ell}$
λ	isotropic length scale in the correlation function for the KL decomposition
λ_i	eigenvalues in covariance eigen-decomposition
Λ	matrix of eigenvalues in covariance eigen-decomposition
M	number of points of a computational domain
\mathbf{M}	ARD diagonal matrix of length scales
M_ℓ	grid at level ℓ
$Mass$	integral of the concentration in the domain D
$m(\mathbf{x})$	the prior mean function of the Gaussian process
$m_{\mathcal{D}}$	the posterior mean function of the Gaussian process
m_j	predicted expected value
$m_{(i)}^*$	the posterior mean function for each random function $f_{(i)}$
\mathbf{m}	mean vector in KL decomposition, $\mathbb{E}[\mathbf{Z}]$
μ	fluid viscosity
$\boldsymbol{\mu}$	mean vector for the Normal distribution
μ_i	i^{th} component of vector $\boldsymbol{\mu}$, $\mu_i = m(\mathbf{x}_i)$
N	number of samples
$\mathcal{N}(\boldsymbol{\mu}, \Sigma)$	Gaussian (Normal) distribution with mean vector $\boldsymbol{\mu}$ and covariance matrix Σ
\mathbb{N}	the natural numbers
N_ℓ	number of samples at level ℓ
\otimes	tensor product operator
Ω	sample space, set of all outcomes for a given experiment
Ω_M	computational domain, set of nodes

\mathbb{P}	probability measure
p	probability
P	fluid pressure
P_{in}	pressure at left boundary
P_{out}	pressure at right boundary
P_j	pressure field j
\tilde{P}_j	reduced rank approximation of pressure field P_j
P^*	emulated pressure field
ϕ	rock porosity
ϕ_i	eigenvector i in covariance eigen-decomposition
Φ	matrix of eigenvectors in covariance eigen-decomposition
Ψ	streamfunction
Ψ_j	streamfunction field j
$\tilde{\Psi}_j$	reduced rank approximation of streamfunction field Ψ_j
Ψ^*	emulated streamfunction field
$\pi(\cdot)$	class probability function
\mathbf{q}	Darcy's flux
R	right singular matrix in the SVD method
R_*	reduced rank approximation od matrix R
Ra	Rayleigh number
\mathbb{R}	the set of real numbers
ρ	fluid density
RE	relative error
s_j^2	variance of the predicted expected value
σ^2	variance in the correlation function for the KL decomposition
σ_d^2	design points variance
σ_f^2	signal variance
σ_n^2	noise variance
$\Sigma(\mathbf{X}, \mathbf{X})$	$d \times d$ covariance matrix
$\Sigma(\mathbf{X}, \mathbf{X}_*)$	$d \times d_*$ covariance matrix between training and test cases
Σ	prior covariance matrix of a GP
Σ_{ij}	entries of the matrix Σ , $\Sigma_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$
SF	surface flux
SF_o	surface flux value for no-flow solution
SF_j	surface flux value for permeability field \mathbf{K}_j , $f(\mathbf{K}_j)$

θ	general vector of hyperparameters
θ_{SE}	vector of hyperparameters for <i>SE</i> covariance function
θ_{Matern}	vector of hyperparameters for <i>Matern</i> covariance function
θ_{RQ}	vector of hyperparameters for <i>RQ</i> covariance function
\mathbf{t}_j	column j of matrix R_*^\intercal
τ_i	observed travel time for permeability field \mathbf{K}_i , $f(\mathbf{K}_i)$
τ_i^*	predicted travel time for permeability field \mathbf{K}_i , $\hat{f}(\boldsymbol{\xi}_i)$
T	(inaccesible) random variable
\hat{T}_M	general estimator of $\mathbb{E}(T_M)$
$\hat{T}_{M,N}^{MC}$	MC estimator
$T_M^{(i)}$	i^{th} sample of T_M
T_M	functional of \mathbf{X}_M , $f(\mathbf{X}_M)$
T_{M_ℓ}	functional of \mathbf{X}_{M_ℓ} , $f(\mathbf{X}_{M_\ell})$
$T_{M_\ell}^{(i)}$	functional of sample i at level M_ℓ , $f(\mathbf{X}_{M_\ell}^{(i)})$
\hat{T}_M^{ML}	general multilevel estimator
$\hat{T}_{M,\{N_\ell\}}^{MLMC}$	multilevel estimator for the MC level estimator with N_ℓ samples
$\hat{T}_{M,N}^{QMC}$	QMC estimator for $\mathbb{E}[T_M]$
$\hat{T}_{M,\{N_\ell\}}^{MLQMC}$	MLQMC estimator for the QMC level estimator with N_ℓ samples
T_{MC}	estimated averaged travel time with the MC method
T_{MLMC}	estimated averaged travel time with the MLMC method
T_{QMC}	estimated averaged travel time with the QMC method
T_{MLQMC}	estimated averaged travel time with the MLQMC method
T_{GP}^{SE}	averaged travel time prediction with <i>SE</i> covariance function
$T_{GP}^{3/2}$	averaged travel time prediction with <i>Matérn</i> $_{\frac{3}{2}}$ covariance function
$T_{GP}^{5/2}$	averaged travel time prediction with <i>Matérn</i> $_{\frac{5}{2}}$ covariance function
T_{GP}^{RQ}	averaged travel time prediction with <i>RQ</i> covariance function
TOL	tolerance for optimum number of eigenvector in the reduced rank approximation

U	uniform random variable
\mathbf{u}	Darcy's velocity
u_x and u_z	velocity in the x and z directions
\mathbb{V}	variance of a random variable
$\omega^{(i)}$	realisation for sample i
ξ_i	independent and identically distributed (i.i.d.) random variables $\mathcal{N}(0, 1)$ (or KL coefficients)
ξ_i	design point i
ξ_i^*	test input i for the emulator
\mathbf{X}	$D \times d$ matrix of the training inputs $\{\mathbf{x}_i\}_{i=1}^d$, the design matrix
\mathbf{X}_*	matrix of test inputs
\mathbf{x}_i	the i th training input
\mathbf{x}_{*_i}	the i th test input
\mathbf{X}_M	random vector that takes values in \mathbb{R}^M
\mathbf{X}_{M_ℓ}	random vector at grid M_ℓ
\mathbf{y}	noisy observations
$y x$ and $p(y x)$	conditional random variable y given x and its probability (density)
Y	$M \times n$ matrix of observed fields used for field emulation
\tilde{Y}	row centered version of matrix Y
$\tilde{\mathbf{y}}_j$	column j of matrix \tilde{Y}
$\tilde{\mathbf{y}}_j^*$	reduced rank approximation of $\tilde{\mathbf{y}}_j$
Y_ℓ	corrections at level ℓ , $Y_\ell = \mathbf{T}_{M_\ell} - \mathbf{T}_{M_{\ell-1}}$
\hat{Y}_ℓ	estimator for the corrections $\mathbb{E}[Y_\ell]$
$\hat{Y}_{\ell, N_\ell}^{MC}$	MC estimator for the corrections at level ℓ with N_ℓ samples
$\hat{Y}_{\ell, N_\ell}^{QMC}$	QMC estimator for $\mathbb{E}[Y_\ell]$
$\mathbf{0}$	vector of all 0's
$\mathbf{z}(t)$	position of a particle at time t
$Z(\mathbf{x}, \omega)$	realisation of a random field

References

- K. F. Bannör and M. Scherer. *Model risk and uncertainty - Illustrated with examples from mathematical finance.* Risk - A Multidisciplinary Introduction. Chapter 10, 279-306, SPRINGER, 2014.
- J. Bear. *Dynamics of Fluids in Porous Media.* New York; London: American Elsevier, 1972.
- D. Bolster, M. Barahona, M. Dentz, D. Fernandez-Garcia, X. Sanchez-Vila, P. Trinchero, C. Valhondo, and D. M. Tartakovsky. *Probabilistic risk analysis of groundwater remediation strategies.* Water Resour. Res. 45, W06413, 2009.
- S. C. Brenner and L. R. Scott. *The Mathematical Theory of Finite Element Methods.* Springer-Verlag New York, Inc, 2002.
- S. Burhenne, D. Jacob, and G. P. Henze. *Sampling based on Sobol Sequences for Monte Carlo techniques applied to building simulations.* 12th Conference of International Building Performance Simulation Association, Sydney, 14-16 November, 2011.
- E. Byers and D. B. Stephens. *Statistical and stochastic analyses of hydraulic conductivity and particle-size in a fluvial sand.* Soil Science Society of America Journal, 47:1072-1081, 1983.
- K. A. Cliffe, A. Spence, and S. J. Tavener. *The numerical analysis of bifurcation problems with application to fluid mechanics.* Acta Numerica, Cambridge University Press, 2000a.

- K. A. Cliffe, M. B. Giles, R. Scheichl, and A. L. Teckentrup. *Multilevel Monte Carlo Methods and Applications to Elliptic PDEs with Random Coefficients*. Comput Visual Sci 14:3-15, Springer-Verlag, 2011.
- K.A. Cliffe, I.G. Graham, R. Scheichl, and L. Stals. *Parallel computation of flow in heterogeneous media using mixed finite elements*. J. Comput. Phys. 164, 258-282, 2000b.
- N. Collier, A-L. Haji-Ali, F. Nobile, E. von Schwerin, and R. Tempone. *A Continuation Multilevel Monte Carlo algorithm*. BIT Numerical Mathematics, 2014.
- D. Cornford, I. T. Nabney, , and C. K. I Williams. *Modelling Frontal Discontinuities in Wind Fields*. Journal of Nonparameteric Statsitics, 14(1-2):4358, 2002.
- J. A. Cumming and M. Goldstein. *Small Sample Bayesian Designs for Complex High-Dimensiona Models Based on Information Gained Using Fast Approximations*. TECHNOMETRICS, VOL. 51, NO. 4, 2009.
- A. P. Dawid and P. Sebastiani. *Coherent disperson criteria for optimal experimental design*. The Annals of Statistics 27, 65(81), 1999.
- G. de Marsily. *Quantitative Hydrogeology*. Academic Press, London, 1986.
- M. Dentz and D. M. Tartakovsky. *Probability density functions for passive scalars dispersed in random velocity fields*. Geophysical Research Letters 37, 2010.
- M. Dentz, P. Gouze, and J. Carrera. *Effective non-local reaction kinetics for transport in physically and chemically heterogeneous media*. Journal of Contaminant Hydrology, 2011a.
- M. Dentz, T. Le Borgne, A. Englert, and B. Bijeljic. *Reactive Transport and Mixing in Heterogeneous Media: A Brief Review*. Journal of Contaminant Hydrology, 2011b.
- C. R. Dietrich and G. Newsam. *A stability analysis of the geostatistical approach to aquifer identification*. Stochastic Hydrol. Hydraul., 4(3), 293-316, 1989.

- C. R. Dietrich and G. Newsam. *Fast and exact simulation of stationary Gaussian processes through circulant embedding of the covariance matrix.* SIAM J. SCI. COMPUT., 18(4), 1088-1107, 1997.
- M. W. Farthing, M. A. Seyedabbasi, P. T. Imhoff, and C. T. Miller. *Influence of porous media heterogeneity on nonaqueous phase liquid dissolution fingering and upscaled mass transfer.* Water Resources Research, Vol. 48, W08507, 2012.
- R. Ghanem and D. Spanos. *Stochastic Finite element: a spectral approach.* Springer, New York, 1991.
- K. Ghesmat, H. Hassanzadeh, and J. Abedi. *The impact of geochemistry on convective mixing in a gravitationally unstable diffusive boundary layer in porous media: CO₂ storage in saline aquifers.* J. Fluid Mech., vol. 673, pp. 480-512, Cambridge University Press, 2011.
- M. Giles and B.J. Waterhouse. *Multilevel quasi-Monte Carlo path simulation.* Radon Series Comp. Appl. Math 8, 1-18, 2009.
- M. B. Giles. *Multilevel Monte Carlo Path Simulation.* Operations Research, Vol. 56, No. 3, pp. 607-617, 2008.
- M. B. Giles, T. Nagapetyan, and K. Ritter. *Multilevel Monte Carlo Approximation of Distribution Functions and Densities.* SIAM/ASA J. UNCERTAINTY QUANTIFICATION, Vol. 3, pp. 267-295, 2015.
- I.G. Graham, F.Y. Kuo, D. Nuyens, R. Scheichl, and I.H. Sloan. *Quasi-Monte Carlo methods for elliptic PDEs with random coefficients and applications.* J. Comput. Phys. 230(10), 3668-3694, 2011.
- J. Hadley. How Monte Carlo simulation fights the curse of dimensionality. <http://decision-analytics-blog.lumina.com/monte-carlo-simulations/how-monte-carlo-simulation-fights-the-curse-of-dimensionality/>, 2013. Lumina.
- D. A. Henderson, K. J. Krishnan R. J. Boys, C. Lawless, and D. J. Wilkinson. *Bayesian Emulation and Calibration of a Stochastic Computer Model of*

- Mitochondrial DNA Deletions in Substantia Nigra Neurons.* Journal of the American Statistical Association, 2009.
- J. Hidalgo and J. Carrera. *Effect of dispersion on the onset of convection during CO₂ sequestration.* J. Fluid Mech. 640, 441-452, 2009.
- R.J. Hoeksema and P.K. Kitanidis. *Analysis of the spatial structure of properties of selected aquifers.* Water Resources Research. 21, 536-572, 1985.
- P. B. Holden, N. R. Edwards, P. H. Garthwaite, and R. D. Wilkinson. *Emulation and interpretation of high-dimensional models.* submitted to Journal of Applied Statistics, 2015.
- P. Houston. Aptofem. Finite element software toolkit, School of Mathematics, University of Nottingham. <https://www.maths.nottingham.ac.uk/personal/ph/Site/Software.html>.
- IPCC(AR5). Intergovernmental panel on climate change. <https://www.ipcc-wg2.gov/AR5-tools/>.
- J.Dick and F.Pillichshammer. *Digital Nets and Sequences. Discrepancy Theory and Quasi-Monte Carlo Integration.* Cambridge University Press, Cambridge, 2010.
- F. J.Hickernell and H.Niederreiter. *The existence of good extensible rank-1 lattices.* J. Complexity 19, 286-300, 2003.
- M. C. Kennedy and A. O'Hagan. *A. Bayesian calibration of computer models.* J. R. Statistical Society Part B, 63(Part 3):425-464, 2001.
- G. J. Lord, C. E. Powell, and T. Shardlow. *An introduction to computational stochastic PDEs.* Cambridge texts in applied mathematics, 2014.
- B. Matérn. *Spatial Variation.* Meddelanden från Statens Skogsforskningsinstitut, 49, No.5. Almänna Förlaget, Stockholm. Second edition (1986), Springer-Verlag, Berlin, 1960.

- M. D. McKay, R.J. Beckman, and W.J. Conover. *A comparison of three methods for selecting values of input variables in the analysis of output from a computer code.* Technometrics, 21(2):239-245, 1979.
- T. P. Minka. *A Family of Algorithms for Approximate Bayesian Inference.* PhD thesis, Massachusetts Institute of Technology, 2001.
- A. Mondal, Y. Efendiev, B. Mallick, and A. Datta-Gupta. *Bayesian Uncertainty Quantification for Flows in Heterogeneous Porous Media using Reversible Jump Markov Chain Monte Carlo Methods.* Advances in Water Resources 33, 211-256, Elsevier, 2010.
- R. M. Neal. *Bayesian Learning for Neural Networks.* Springer, New York. Lecture Notes in Statistics 118, 1996.
- J. A. Neufeld, M. A. Hesse, A. Riaz, M. A. Hallworth, H. A. Tchelepi, and H. E. Huppert. *Convective dissolution of carbon dioxide in saline aquifers.* Geophys. Res. Lett., 37, L22404, 2010.
- J. Oakley and A. O'Hagan. *Bayesian inference for the uncertainty distribution of computer model outputs.* Biometrika, 89, 4, pp. 769-784, 2002.
- A. O'Hagan. *Bayesian Analysis of Computer Code Outputs:A Tutorial.* University of Sheffield, UK, 2004.
- E. J. Pebesma and G. B. M Heuvelink. *Latin hypercube sampling of gaussian random fields.* Technometrics, 41(4):303-312, 1999.
- P. Ranganathan, R. Farajzadeh, H. Bruining, and P. L. J. Zitha. *Numerical Simulation of Natural Convection in Heterogeneous Porous media for CO₂ Geological Storage.* Springerlink.com, 2012.
- C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning.* the MIT Press, 2006.
- D. Russo and M. Bouton. *Statistical analysis of spatial variability in unsaturated flow parameters.* Water Resources Research, 28(7):1911-1925, 1992.

- P. G. Saffman. *A theory of dispersion in a porous medium.* Journal of Fluid Mechanics, Cambridge University Press, 1959.
- A. Saltelli, P. Annoni, I. Azzini, F. Campolongo, M. Ratto, and S. Tarantola. *Variance based sensitivity analysis of model output. Design and estimator for the total sensitivity index.* Computer Physics Communications, 181(2):259-270, 2010.
- A. E. Scheidegger. *General theory of dispersion in porous media.* Journal of Geophysical Research, American Geophysical Union, 1961.
- I. M. Sobol. *Distribution of points in a cube and approximate evaluation of integrals.* U.S.S.R Comput. Maths. Math. Phys. 7: 86-112, 1967, 1967.
- M. L. Stein. *Interpolation of Spatial Data.* Springer-Verlag, New York, 1999.
- N. Stone. *Gaussian Process Emulators for Uncertainty Analysis in Groundwater Flow.* PhD Thesis. University of Nottingham, 2011.
- G. Strang. *Introduction to linear algebra.* Cambridge (MA): Wellesley-Cambridge Press, 2003.
- A. M. Tartakovsky, D. M. Tartakovsky, T. D. Scheibe, and P. Meakin. *Hybrid Simulations of Reaction-Diffusion Systems in Porous Media.* SIAM J. Sci. Comput., 30(6), 27992816, 2007.
- T. J. Ward, K. A. Cliffe, O. E. Jensen, and H. Power. *Dissolution-driven porous-medium convection in the presence of chemical reaction.* J. Fluid Mech., vol. 747, pp. 316-349, Cambridge University Press, 2014.
- R. D. Wilkinson. *Computational Methods for Large-Scale Inverse Problems and Quantification of Uncertainty.* People on Earth, John Wiley & Sons, Ltd, 2011.
- R. D. Wilkinson, M. Vrettas, D. Cornford, and J. E. Oakley. *Quantifying simulator discrepancy in discrete-time dynamical simulators.* Vol. 16, Issue 4, pp 554-570. Journal of Agricultural, Biological, and Environmental Statistics, 2011.

Y. Xie, C. T. Simmons, A. D. Werner, and H-J. G. Diersch. *Prediction and uncertainty of free convection phenomena in porous media.* Water Resources Research, Vol. 48, W02535, 2012.