Scientific Research Publishing

Open Journal of Modelling and Simulation, 2022, 10, 349-390
https://www.scirp.org/journal/ojmsi
ISSN Online: 2327-4026
ISSN Print: 2327-4018

Quasi-Monte Carlo Approximations for Exponentiated Quadratic Kernel in Latent Force Models

Qianli Di

Department of Computer Science, University of Sheffield, South Yorkshire, UK
Email: wanghui_6557@yeah.net

How to cite this paper: Di, Q.L. (2022) Quasi-Monte Carlo Approximations for Exponentiated Quadratic Kernel in Latent Force Models. *Open Journal of Modelling and Simulation*, **10**, 349-390.
https://doi.org/10.4236/ojmsi.2022.104021

Received: June 24, 2022
Accepted: October 10, 2022
Published: October 13, 2022

Abstract

In this project, we consider obtaining Fourier features via more efficient sampling schemes to approximate the kernel in LFMs. A latent force model (LFM) is a Gaussian process whose covariance functions follow an Exponentiated Quadratic (EQ) form, and the solutions for the cross-covariance are expensive due to the computational complexity. To reduce the complexity of mathematical expressions, random Fourier features (RFF) are applied to approximate the EQ kernel. Usually, the random Fourier features are implemented with Monte Carlo sampling, but this project proposes replacing the Monte-Carlo method with the Quasi-Monte Carlo (QMC) method. The first-order and second-order models' experiment results demonstrate the decrease in NLPD and NMSE, which revealed that the models with QMC approximation have better performance.

Keywords

Latent Force Model, COVID-19, Quasi-Monte Carlo Approximations

1. Introduction

This project considers improving the performance of random Fourier feature maps to speed up the testing and training speed when dealing with the large data sets in LFMs. Besides, this project proposes replacing the MC approximation with the QMC approximation, a more efficient sampling scheme for the kernel approximation.

Latent force model (LFM) [1] is a type of Gaussian process with multiple-output. LFM can be used to reveal the dynamics in the gene network [2], to infer human movements from motion capture data [3], for motion primitive segmentation in humanoid robot [4], and it can also be used for modelling the buildings'

thermal characteristics [5]. Among several other applications mentioned above, the prior knowledge of the mechanical model can be encoded with the covariance function of the Gaussian process. After combining the physics with the covariance function of a GP, the extrapolation ability can be granted to another interpolation-only model.

In classical LFMs, the expression of the covariance function $k(t,t')$ is in the form of Exponentiated Quadratic (EQ) that leads to the solutions for the cross-covariances $k_{fd,fd'}(t,t')$. While obtaining the solutions, the cost is expensive because the solutions include complex calculations that can be obtained only by numerical methods. To easier the expressions involved in the LFM's covariance functions, random Fourier features (RFF) [6] are applied to approximate the calculation of EQ kernel with Monte Carlo method [3].

$$k_q(T,T') = \int \mathrm{e}^{j(T,T')\omega} p(\omega)\mathrm{d}(\omega) \tag{1.1}$$

The expression of $k_{fd,fd}(t, t)$ is simplified as

$$\sum_{q=1}^{Q} \frac{S_{d,q}S_{d',q}}{S}\left[\sum_{S=1}^{S} v_d(t,\theta_d,\lambda_s)v_d(t',\theta_{d'},\lambda_s)\right] \tag{1.2}$$

Therefore, the double integrals are transformed into two separate integrals which reduce the cost of computation.

For the QMC method, there are modifications can be made in the expression $\int \mathrm{e}^{j(\tau-\tau')w} p(w)\mathrm{d}(w)$, the sampling points generated by w with the traditional MC method is randomly distributed. Different from the MC method, the QMC method will choose the points generated in a unit cube, where the points are firstly generated in discrepancy sequence $t_1,\cdots,t_s \in [0,1]^d$. Then, a cumulative function of p_j, Φ_i will be used to transformed the sequence. Through setting $w_i = \Phi^{-1}(t_i)$, and $\Phi^{-1}(t) = (\Phi^{-1}(t_1),\cdots,\Phi^{-1}(t_d)) \in R^d$, the integral $\int \mathrm{e}^{j(\tau-\tau')w} p(w)\mathrm{d}(w)$ can be converted over the unit cube.

This project aims to get the random Fourier feature maps via replacing the MC method with the QMC method. Then, the LFMs with new approximated kernels will be applied to carry out the experiments. Also, this project will use NLPD and NMSE as evaluation metrics.

1.1. Aims and Objectives

This project aims to find a more efficient sampling scheme to obtain the random Fourier features and improve the efficiency of random Fourier feature maps to speed up the EQ kernel's training and testing rate in LFMs. Furthermore, this project considers using the QMC method to replace the MC method and implementing 4 kinds of low-discrepancy sequences to assess the performance.

1.2. Chapter Overview

Chapter 2—Literature review: This chapter will firstly review relative background knowledge. It begins with the study of latent force models and kernel methods. Then, it goes to the introduction of random Fourier features. Finally,

this project will introduce the MC and QMC methods, the sampling schemes applied.

Chapter 3—Requirements: begins with the analysis of the detailed requirements of this object.

Besides, ethical issues, evaluation are discussed in the section.

Chapter 4—Design: It contains the overall design and the basic models. Besides, the detailed datasets for experiments and the low discrepancy sequences are also discussed in this section.

Chapter 5—Implementation and testing: The steps to implement the projects, for example, the preprocessing of data, the training of models, the implementation of sequences, will all be discussed in this part.

Chapter 6—Experiments and discussion: Two different datasets are used to experiments with the approximated model with QMC methods. The experimental tables and figures are posted out. This project judges the performance according to the metrics mentioned in chapter 3. Moreover, the goals achieved by the project and the future work are discussed in this section.

Chapter 7—Conclusion: The findings and improvements of this project are discussed in this chapter.

1.3. Discussion

The project is based on machine learning and numerical analysis. According to the courses of Master degree, the lesson about machine learning is provided, and it is the theoretical basis of this project. Besides, the practices and assignments of the machine learning course supply me with experience dealing with data and basic sampling technology.

However, the experience and knowledge gained from the course cannot offer a solution to this project. The project needs a solid foundation of mathematics and Matlab. Moreover, students need to have a clear understanding of how the physic models work, and the basic knowledge of algorithms is required.

It is a challenge for a traditional computer science student, but it is also a chance for students to master numerical analysis skills.

2. Literature Review

This section will first give a general review of the latent force models. Then, the following two parts will review the kernel methods and random Fourier features (RFF), which are the key points to developing our project. Finally, the last two parts introduce the two different sampling methods, the Monte-Carlo method, which is classic, and the Quasi-Monte Carlo method (QMC), which is applied in this project.

2.1. Latent Force Models

2.1.1. Gaussian Process

The latent force model (LFM) belongs to the Gaussian process. Gaussian Process [7] is a concept in statistics, and it belongs to a particular example in the Sto-

chastic process. It's a set of random variables in a continuous domain, with each random variable obeying a Gaussian distribution at each time or space point. A Gaussian Process is decided by the mean and covariance function. The characteristics are controlled by the kernel function, which denotes the similarity of pairs of points [8]. The GP is a powerful model that can be used to represent the distribution of a function. It directly models the function to generate a non-parametric model. One of the outstanding advantages is that it can simulate any black-box functions and simulate uncertainty. As for the covariance function, this part is widely discussed later in the application of machine learning, and it is also called the kernel function. The reason is that it captures the relationship between different input points and reflects the position of the subsequent samples. Gaussian processes can be entirely specified by second-order statistics, which is a fundamental feature. Therefore, if the mean value of the Gaussian process is regarded as zero, the covariance function formula can fully explain the behaviour of the process [9].

2.1.2. Introduction of Latent Force Model

LFM is a Gaussian process having multiple outputs. It has the feature that its covariance function contains ordinary differential equations (ODEs) or partial differential equations (PDEs). Especially, every output in the LFMs can be described as $D_d\left\{f_d(t)\right\}=u(t)$, in which D_d is the differential operator related to a partial or ordinary differential equation. Besides, $u(t)$ is the excitation function. A latent force model presumes that $u(t)$ is unknown, and a Gaussian process is placed over it. The expression for $f_d(t)$ can be written as

$$f_d(t)=\int_0^t G_d(t-T)u(T)\mathrm{d}T$$

in which $G_d(\cdot)$ is equivalent to the Green's function which is related to the differential operator D_d. The function $u(t)$ or the latent force is not observed, and it follows a GP prior having mean function zero, and covariance function given by $k(t, t)$. After $u(t)$ being modified by the linear operator, the expression of $f_d(t)$ also follows a Gaussian process having covariance $k_{fd,fd0}(t, t)$. Moreover, there is a possibility to compute the cross-covariance function from $f_d(t)$ and $f_d'(t')$ because every $f_d(t)$ has a $u(t)$ as its input. The equation of $f_d(t)$ can be extended, after containing the extra latent functions with distinct traits, the expression of each output can be written as

$$f_d(t)=\sum_{q=1}^{Q}S_{d,q}\int_0^t G_d(t-T)u(T)\mathrm{d}T$$

in which the Q represents the number or latent forces, and $S_{d,q}$ is a sensitivity parameter which represents the affect of force $u_q(t)$ with output d. Presuming the independence of $u_q(t)$, and they all follow the same covariance $k_q(t, t)$, therefore the cross-covariance functions can be computed [3].

$$k_{fd,fd'}(t,t')=\sum_{q=1}^{Q}S_{d,q}S_{d',q}\int_0^t G_d(t-T)\int_0^{t'}G_d(t'-T')\times k_q(T,T')\mathrm{d}(T)\mathrm{d}(T') \quad (2.1)$$

Based on the expressions for $k_q(t, t')$, a similar expression for $k_{fd,fd0}(t, t')$ may also be found. There is a normal expression for $k_q(\tau, \tau')$ which follows an EQ form.

$$k_q(T,T') = \exp\left[-\frac{(T-T')^2}{\ell_q^2}\right]$$

in which ℓ_q^2 is defined as the length-scale.

This project will further explain the essential concepts related to LFMs, such as kernel methods, the random Fourier features, and the sampling methods.

2.2. Kernel Methods

Considering the EQ kernel in LFMs, there is necessary to introduce the kernel methods. The kernel method is a kind of machine learning method. The so-called machine learning (ML) refers to the use of computers to simulate or realise the learning behaviour of humans to acquire knowledge and experience and to organise new knowledge structures to continuously improve them The performance of the system itself [10]. People's attention to kernel methods [11] [12] [13] benefits from the theoretical and applied development of Support vector machine (SVM) [14] [15]. The adoption of the kernel function makes it easy to extend the linear SVM to the nonlinear SVM. Its core lies in the use of relatively simpler kernel function calculations, which not only avoids the complex inner product calculation in the feature space but also avoids the design of the feature space itself [16] [17]. The research on the kernel function started very early. In 1964, Aizermann *et al.* proposed introducing the kernel method into machine learning when studying the potential function. However, due to the constraints of the limited technology at the time, its advantages did not attract widespread attention [18]. The kernel method effectively solves the problem of high-dimensional model construction under limited samples and has a strong generalisation ability. The theoretical foundation of nuclear methods is ancient. Mercer's theorem, which has a cornerstone function, was found in 1909. The research on Reproducing Kernel Hilbert Space (RKHS) began in the 1940s [19]. Aronszajn's detailed exposition on RKHS completed in 1950 laid a solid foundation for later development [20]. Kernel methods [21] [22] [23] provide an allinclusive collection of non-parametric modelling skills for problems about machine learning.

2.2.1. The Principles of Kernel Methods

Given a supervised machine learning problem, a sequence of pair points, $(x_1, y_1), (x_2, y_2), \cdots, (x_i, y_i) \in X \times Y$, in which input domain $X \subseteq R^n$, output domain $Y \subseteq R$ (regression problems) or $Y = -1, +1$ (classification problems). A new feature space F can map the input points into via a non-linear mapping, in which $F \subseteq R^n$.

$$\begin{aligned} \psi &: x \to F \\ x &\mapsto \psi(x) \end{aligned} \tag{2.2}$$

Then, a new expression of the data will be used to describe the origin problem

$$\left(\psi\left(x_1\right), y_1\right), \left(\psi\left(x_2\right), y_2\right), \cdots, \left(\psi\left(x_i\right), y_i\right) \in F \times Y \tag{2.3}$$

The kernel function combines the two steps of nonlinear mapping and the inner product of two vectors in the feature space, so that the nonlinear mapping is performed implicitly, and the linear hyper-plane can be composed of all training samples and a test sample in the feature space. The linear combination representation of product terms, thus avoiding the curse of dimension.

According to Mercer condition [24], assume X be the Compact subset of R^n, $k: \chi \times \chi \to R$ is a continuous symmetric function, if its integral operator on the Hilbert space satisfies the integral positive definite condition [25]:

$$\forall f \in L_2(x), \int_{x \times x} k(x,z) f(x) f(z) \mathrm{d}x\mathrm{d}z \tag{2.4}$$

Then, there must be a feature space F and a mapping ψ, let

$$k(x,z) = \psi(x) \times \psi(z) \tag{2.5}$$

Different kernel functions can be designed for different uses. Linear kernels, polynomial kernels, radial basis kernels, Sigmund kernels, Fourier kernels, and other kernel functions are extensively utilised. In particular, if a kernel function k is a function of $(x - y)$, it is called a shift-invariant kernel. The Gaussian kernel, Laplace kernel, and the EQ kernel in LFM belong to this type of kernel. In space F, normal regularised linear statistical models offer the origin input expression nonlinear inference. Classical Representer Theorems [22] [23] contains these basic algorithms of such constructions, and the finite-dimensional solutions of related optimisation problems will be promised although F is an infinite-dimension space.

2.2.2. How Feature Maps Approximate the Kernels

Let us review the randomised creation of low-dimensional approximate feature maps [6] which is used to scale up the kernel methods. Feature maps, $\hat{\psi}: \chi \to C^s$, provide the kernel function $k: \chi \times \chi \to C$ with low-distortion approximations:

$$k(x,z) \approx \left\langle \hat{\psi}(x), \hat{\psi}(z) \right\rangle c^S \tag{2.6}$$

in which C^s represents the s-dimensional space with complex numbers and the inner product, with z^* representing the complex number z's conjugate. Although the real-valued feature map is defined [6], Jiyan's technical description is predigested by using the generality of complex-valued features [26]. The expressions in (2.1) leads to solutions, e.g., for problems like regression problems, we have $O(ns^2)$ training and $O(s + maptime)$ predictions speed in which the maptime means it should take the time to create features for test inputs, with $O(ns)$ memory requirements. Especially, the approximation mentioned in the Equation (2.10) is useful to a kind of significant kernels which is called shift-invariant kernel. Assume that a kernel function K on R^d, $K(x, z) = f(x - z)$, the kernel will be defined as shift-invariant kernel, for some complex-valued positive definite

functions f on R^d. Positive definite function is a function which satisfy the requirement that if it is given a set of n points, $y_1 \cdots y_n \in R^d$, the $n \times n$ matrix B defined by $B_{ij} = g(y_i - y_j)$ is positive semi-definite.

From the above review of kernel methods, we can know that the data can be mapped to the-high dimensional space by kernel function, and we only need to calculate the $k(x, y)$ to realise the learning algorithm, with no need to know the implicate feature mapping function $\varphi(\cdot,\cdot)$, but this convenience may lead to some problems. For instance, if we take consideration into realising a non-linear classification algorithm, and the data set is large, thus, it will take much time and space cost to calculate the kernel function, due to the factor that $k(\cdot,\cdot)$ calculates a value for (x, y) at each point, and finally forms a kernel matrix whose size is the square of the data amount. Therefore, the calculation cost should be under consideration when the kernel function is non-linear and with big data sets.

2.3. Random Fourier Features

Rahimi and Recht (2007) paid attention to the above problems, and propose an estimation method based on Fourier transformation [6]. In simple, RFF maps two pieces of origin data x, y to a low-dimensional Euclidean space R^D via an explicit mapping function Z, that is Z: $R^d \to R^D$, and the inner product between the x and y point after mapping will be the approximated value of the kernel function $k(x, y)$:

$$k(x,z) = \langle \phi(x), \phi(y) \rangle \approx Z(x)^{\mathrm{T}} \cdot Z(y) \tag{2.7}$$

Z function maps x to a relatively low-dimension space which is different from the feature mapping function $\varphi(\cdot)$ in the function $k(\cdot,\cdot)$ will map the x to a high-dimension space. Hence, we can use function Z to transform the input data, and then apply linear methods to approximate the non-linear algorithms.

According to the Bochner [27] theorem, a function f which is complex-valued, and it is positive definite if and only if the function is the Fourier Transform of a Borel measure μ which is finite non-negative on R^d,

$$f(x) = \hat{\mu}(x) = \int_{R^d} \mathrm{e}^{-ix^{\mathrm{T}}w} \mathrm{d}\mu(w), \forall x \in R^d \tag{2.8}$$

With no loss of generality, this project suppose that $\mu(\cdot)$ is a probability measure and the function has its related density function $p(\cdot)$. As a result of the expression above, a scaled kernel that is shift-invariant could have a one-to-one correspondence with the density function p, implying that,

$$k(x,z) = f(x-z) = \int_{R^d} \mathrm{e}^{-i(x-z)^{\mathrm{T}}w} p(w) \mathrm{d}(w) \tag{2.9}$$

Therefore, we only need to sample W from $p(w)$ om theory in order to approximate the value of $k(x - y)$.

The equation (2.8) can be approximated as:

$$k(x,z) = \int_{R^d} \mathrm{e}^{-i(x-z)^{\mathrm{T}}w} p(w) \mathrm{d}(w) \approx \frac{1}{s} \sum_{j=1}^{s} \mathrm{e}^{-i(x-z)^{\mathrm{T}}} w_s = \langle \hat{\psi}_s(x), \hat{\psi}_s(z) \rangle c^s \tag{2.10}$$

and in which the feature map is:

$$\hat{\psi}_s(x) = \frac{1}{\sqrt{S}}\left[e^{-ix^T w_1} \cdots e^{-ix^T w_s}\right] \in C^s \tag{2.11}$$

The subscript S in the expression, represents the feature map's dependence on the sequence $S = (w_1, \ldots, W_s)$. As the elements in the sequence are obtained from the function $p(\cdot)$, the approximation in expression (2.9) can be thought as Monte Carlo method. Therefore, in the next part this project will introduce the Monte Carlo (MC) method.

2.4. Monte Carlo Method

2.4.1. The Introduction of the Monte Carlo Approach

In the 1940s, members of the world-famous USA-Los Alamos National Laboratory, John von Neumann, Manhattan Project, Stanislaw Ulam and Nicholas pioneered the Monte Carlo method. At that time, this method was mainly applied for nuclear weapon's development and production [28]. However, due to roulette, a simple random number generator, Ulam's uncle is super fond of gambling. Therefore, he always plays in Monte Carlo, and he always loses his cash in Monte Carlo. Hence the approach is called Monaco's world-renowned casino-Monte Carlo [29]. Thus, the name and system development of the Monte Carlo approach began.

There were some isolated and unexplored examples in an earlier period of the MC method. For instance, people randomly throw a needle on a parallel and straight ruler, and they deduced the value of π by looking at the nodes of the needle and the parallel lines. In 1873, a report about an exciting experiment appeared in the paper entitled "Experimental Determination of Pi". There is related software that is developed for simulation experiments by the author. In 1899, Lord Rayleigh also indicated that one-dimensional random walk kinetic energy with no absorption limits approach a parabolic differential equation. In 1931, Kolmogorov also indicated Markov the unique association between the integral equations [30].

The British Statistics Schools sunk in heavy simple Monte Carlo work in the early twentieth century. The majority of the results were like a predicant and rarely applied for researches or discoveries. Under rare circumstances, an original discovery was highlighted instead of comfort validation. For example, in 1908, a student used a sampling experiment to assist him in completing the study of distributing correlation parameters. In the meantime, the student supported the so-called T distribution originated from his shaky and synsemantic theoretical discussions. Nevertheless, the study of developing atomic bombs in the Second World War is the origin of the actual use of the Monte Carlo method as a study device. There is a direct emulation of the probability of randomly diffusing neutrons in nuclear fuel in this work. In the early stages of these tests, this unique "Russian Roulette" and "splitting" approach was refined by von Neu-

mann and Ulan. By 1948, based on Schrodinger's equation, Fermi, Metropolis, and Ulan evaluated its eigenvalues with the MC method, and since then, the study of the MC method has been started.

As we understand it today, there is no statistical sampling technology to seek immediate solutions to quantitative problems in the Monte Carlo approach. Ulam did not propose statistical sampling technology. However, quantitative problems during physical processes were analysed with the statistical sampling technology long ago. Ulam's contributed to identifying the hidden automatic sampling role of the newly developed electronic computer. He developed computer-executed algorithms and investigated the methods of shifting non-random issues into arbitrary forms by statistical sampling. This research changed the statistical sampling method from a purely mathematical approach to a formal process. As a result, the methodology applies to various problems [30].

The Monte Carlo method solves issues by producing suitable random numbers and investigating some given nature or attributes of the information. When carrying out sampling experiments, the MC method provides an approximation for different mathematical issues. This method is very effective for obtaining digital solutions for some issues that are too complex to be analysed and solved. It is also appropriate for issues with no probability and issues with an inborn probability framework.

Monte Carlo simulation technology had officially existed since the early 1940s [30], and it was applied to nuclear fusion research then. As a result, MC has been extensively applied, particularly at the end of the 20th century. As electronic computers develop quickly, the realisation, growth, and enhancement of approaches have been significantly promoted. Monte Carlo methods are emerging endlessly because modern computers can perform millions of simulations more quickly and effectively, which is also an essential element for Monte Carlo simulation to offer suitable solutions fast and guarantee a higher-level precision. After all, it represents that the technology can offer more simulations. For sure, similar solutions are more precise. However, an approximate answer can only be provided by these approaches. Hence, the discussion on the approximate mistake is the major element to be taken into account when using these methods to evaluate the response. Due to the attempt to decrease this approximation mistake, there are various types of Monte Carlo approaches. Different approaches have various extents of precision for their solutions, and the precision level of several approaches will differ with the issue.

Compared with conventional mathematical-statistical approaches, the Monte Carlo approach decreases the difficulty of implementation in virtue of computers. It is hard or even impossible to deal with complex problems with other mathematical approaches. However, an appropriate way can be observed to cope with it. Generally, for the probability allocation of an unclear entity, a simulation is operated many times. The Monte Carlo approach aims at calculating compli-

cated integrals, especially various integrals with almost no study approach. Due to their sophistication, they are appropriate for seeking a suitable solution instead of calculating the complex integrals. Under these cases, Monte Carlo's approximation is a useful device because a reasonable approximation can be given faster than other standard technologies. The uses of the Monte Carlo approaches aim at optimisation, numerical integration and probability distribution production.

The MC method is often applied while settling physical and mathematical issues. For example, if it is hard or impossible to get analytical expressions, or it is impossible to use conclusive algorithms, the MC method is critical and most valuable.

The Monte Carlo methods have become a widely promoted and irreplaceable computing tool as electronic information technology develops quickly in statistics and other areas. Especially while settling exploration issues, including optimisation and integration, more immeasurable value is even shown. From econometrics, genetics to computational physics, the Monte Carlo approach approximation [31] derive all kinds of statistics.

2.4.2. The Monte Carlo Method's Essential Concept

The statistical approach is a famous mathematical approach recently. Monte Carlo approach is a statistical experiment approach, primarily through statistical sampling experiment to provide approximate solutions to various mathematical problems, sometimes also known as Random sampling technology. They can be easily defined as statistical simulation approaches. In a statistical simulation, a set of random numbers are adopted to simulate experiments. Hence, the Monte Carlo approach collects various approaches with the same processes. This process includes using random numbers to perform many simulations and the probability of obtaining a similar solution to an issue. Thus, random numbers are the defining characteristic of MC technology in the simulation process.

From the theories of great numbers and moment calculation, when the sample scale value T is infinite, the mean value of the sample in the statistical sampling

$$\frac{1}{T}\sum_{t=1}^{T} h(p_t), \left(p_t \sim f(p)\right) \tag{2.12}$$

will converge to $E_f[h(p)]$'s expectation. In statistical experiments, using the expectation $E_f[h(p)]$ to refer to the mean is not uncommon in statistical algorithms. It exists in many approximate approaches for settling integrals. The Monte Carlo approach to solve the approximate value is based on the methods mentioned above. In addition, this approximation has the same precision as statistical estimation, usually denoted as $o(sqrt(T))$. Hence, Once a sample $p_1, \ldots, p_T$ has been generated using the same probability allocation density function f, all standard statistical devices, such as bootstrapping procedures, apply to this sample. As the above formula shows, when a single Monte Carlo experiment concludes the

output and estimated value, it is impossible to settle the variability. Nevertheless, based on Monte Carlo thought, analysis and study approaches can be straightforward. Moreover, systematic implementation of statistical thought can interpret why the Monte Carlo method performs better than a numerical method to a certain extent.

Concerning various issues, an appropriate solution can be found by the Monte Carlo method. However, MC technology's real solution falls into two categories: the association between the thing itself and the random processes.

The first is a deterministic mathematical issue. While applying the Monte Carlo approach to answer such questions, a related probability model can be constructed based on the problem and the deterministic issue's solution, and it is precisely equivalent to the probability allocation of the established model.

Then, sampling research applies a lot of random experiments to this probability model. A lot of random numbers and variables will be generated, and in the end, the variables determined by the model will be sampled. The arithmetic means nearly equivalent to the approximated value of the deterministic issue.

The second is randomness issues. Unexpected issues are universal in life. For instance, neutron diffusion and other problems belong in the medium because certain deterministic factors affect neutrons. Nevertheless, several uncertain and random elements exert more influence. This issue can be defined as some particular function equations or specific multiple integrals sometimes and can be further changed into a random sampling approach for calculation. However, we seldom apply this kind of indirect simulation approach. On the contrary, a simulation approach is applied based on the probability regulations of actual physical characteristics, applying scientific computers to simulate random sampling experiments. On the contrary, the simulation method uses scientific computers to simulate random sampling experiments with reference to the physic properties.

As we concluded above, the specific problem-solving steps for the two kinds of problems are as follows:

1) Show the probabilistic process and set up a simple and easy-to-use probability model or random model according to the question asked. If the question has random features, the probability process can be described and simulated correctly. While, for the inherently non-random problems, we need to understand the physical process and geometric nature of actual objects to settle this kind of problem by the Monte Carlo method. A probabilistic process can be artificially constructed, and the required solution with the statistical values of the coefficients in the established models can be made. The established model should also conform to actual problems concerning primary characteristic coefficients. 2) Taking a sample from a population with a well-defined distribution. What strategy should be used to select a population with a precise distribution adequately based on the distribution of every random variable in the model should be evaluated, and suitably random numbers in a computer simulation

should be generated. It can be recognised as being made up of the probability distribution of related random variables. As a result, the standard technique is to generate random numbers using a uniform distribution, then follow a provided distribution based on real-world examples before completing the random simulation test.

3) Determination of the estimator. The simulation will be carried out according to the constructed model. While realising the simulation, it is necessary to determine a random variable. Then, many repeated experiments must be done to calculate random solutions to the problem. If the solution to the problem is the expected value of this random variable, the estimator is known as an unbiased estimator.

2.4.3. Random Numbers and Pseudo-Random Numbers

The Monte Carlo method is a means of statistical sampling experiments based on computer-generated random numbers. Random numbers play an immeasurable role in dealing with problems. While applying the Monte Carlo approach to a simulation experiment, random variables following different probability distributions can be generated. The standard random variables are uniformly distributed from 0 to 1 in the interval of random variables in different distributions. The sampling value of this kind of random variable is usually called random numbers, and random numbers are used to sample other distributed random variables.

Now we will introduce two methods for generating uniform random numbers: one of them is called the physic method, which use physical phenomena to generate completely random numbers, such as throwing a sieve, using an optional electronic wheel to generate random numbers with random pulses, but this method has a significant loophole, where random processes cannot be saved for information verification and sometimes the cost of experiments is expensive. The second method is called the mathematical method, where mathematical processes performed by iterative formulas will generate a series of numbers. Although this kind of method relies on a computer, it is easy to implement and time-saving. Nowadays, most methods belong to this type.

In daily Monte Carlo simulation experiments, random numbers are obtained by mathematical approaches and superimposed applications based on explicit recursive formulas. However, mathematically, this kind of random number is not a truly random number. Once it is given an initial value, a random number sequence will be determined. Hence, the random number produced by this approach is usually known as pseudo-random numbers.

Nevertheless, these pseudo-random numbers can pass every rigorous test in actual simulation experiments.

Therefore we can use pseudo-random numbers instead of random numbers.

2.4.4. The Characteristics of the Monte Carlo Method

In this subsection, we will review the Monte Carlo integration about its conver-

gence and error. Assume that when computing an approximation of the following expression,

$$I_d[f] = \int_{[0,1]^d} f(x)\mathrm{d}x \tag{2.13}$$

We can notice that when X_d is a random variable which is distributed uniformly in the interval $[0, 1]^d$, then $I_d[f] = E[f(X_d)]$. By drawing a sequence containing random points $S = w_1, \ldots, W_s$ independently from the interval $[0, 1]^d$, then we can compute:

$$I_S[f] = \frac{1}{s}\sum_{w\in S} f(x) \tag{2.14}$$

This process is called the MC method. Then, we define the integration error about the sequence S as

$$\in s[f] = \left|I_d[f] - I_S[f]\right|. \tag{2.15}$$

In the above equation, as the sequence S is drawn randomly, according to the Central Limit Theorem [32], for s large,

$$\in s[f] \approx \sigma[f]s^{-1/2}v \tag{2.16}$$

in which v is a stand normal random variable, and variance f's square-root is $\sigma[f]$. That is,

$$\sigma[f] = \left(\int_{[0,1]^d} f(x) - I[f]^2\,\mathrm{d}x\right)^{1/2} \tag{2.17}$$

The above expression means the error of the Monte Carlo approach converges at the rate of $O(S^{-1/2})$.

2.4.5. Advantages and Disadvantages

The advantages of the MC approaches are listed below. Based on the explanation of the elementary concepts and features of the Monte Carlo approach, it is not confined to mistakes and convergence. Moreover, the MC method has unique features compared with the general mathematical calculation approach. The advantages of the MC methods will be discussed in the following fields:

1) The Monte Carlo approach is intuitive and easy to know. Monte Carlo analysis of problems establishes probabilistic models for physical tests based on the actual physical features of the issue itself. Moreover, it shows the features of describing the randomness of things. While settling actual issues, the Monte Carlo approach begins with the issue itself. Thus, it directly establishes the model rather than construct numerical equations or complex mathematical expressions. Therefore, it guarantees its intuitive and straightforward characteristics. 2) Underestimation of the extensive adaptability of the Monte Carlo approach shall not be made. The prominent characteristic is that while settling issues, the impacts of conditional constraints are minimal. For instance, while calculating various integrals in any area in any dimensional space, for any unusual shape of the integration region, the Monte Carlo approach can produce a lot of uniformly

distributed points in the integration region via the computer. Next, the approximate value of the integration can be obtained by analysing and calculating the experimental outcomes. In the case of very irregular or complicated integration regions and challenging numerical methods to calculate, the Monte Carlo approach is especially remarkable [33].

3) Probability convergence is not related to the dimension of the issue. Based on the previous explanation of the error, when the confidence level remains the same, the convergence speed of the Monte Carlo method is $O(1/sqrt(S))$, which has nothing to do with the dimension of the issue itself. Furthermore, as dimensionality changes, the sampling time and the calculation time of the estimator can be increased with no influence on the error. Namely, in the Monte Carlo approach experiment, the number of samples N is not associated with the dimension S. A growth of the dimension S will not alter the original mistake of the issue but will only cause an increase in the amount of calculation. Based on these advantages, the Monte Carlo method is more desirable and applicable when solving high-dimensional problems than the usual numerical methods.

Here we conclude the disadvantages of the MC method. In conclusion, the MC method shows the convergence rate of $O(1/sqrt(S))$, High-precision approximate outcomes are usually hard to get.

By comparing with other approaches, the efficiency may not be excellent while solving the low dimensionality problems. Meanwhile, the estimation of the mistake of the MC approach is applied at a specific confidence level, which differs from the traditional numerical error computation. As a result, the error will vary depending on the level of confidence. Under general circumstances, for an approximate solution with a certain degree of precision, the MC method needs many experiments, which increases the amount of calculation and decreases the efficiency of the computer (See **Figure 1**).

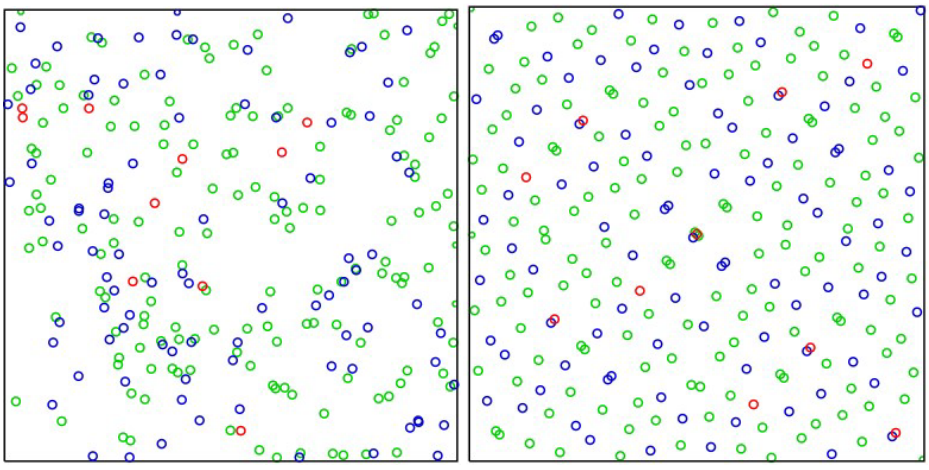

Figure 1. Comparison between MC and QMC.Reproduced from https://en.wikipedia.org/wiki/Main_Page/wiki/Sobol sequence with the permission of the copyright owner.

2.5. Quasi-Monte Carlo Sampling Method

2.5.1. Introduction of the QMC Methods

A typical component in computer applications is efficiently generating random integers uniformly distributed in high-dimensional space. A uniformly distributed random number means a superior sample distribution for all algorithms that need to be sampled. Financial mathematics and computational mathematics more apply quasi-Monte Carlo approaches [34]. The Quasi-Monte Carlo approach aims to solve the problem associated with numerical integration and several othe, the QMC method shows a quicker rate [34].

QMC approaches refer to use a deterministic low-discrepancy sequence to establish S rather than randomly sampling points. **Figure 1** illustrates the underlying intuition as it shows a set of 256 2D random points (left graph) and 256 2D points from a Sobol sequence (right graph). In the random sequence, n undesired clustering of points and, consequently, empty spaces can be seen. Therefore, clusters add little to the integral approximation in those areas, while the sampling of the integrand is not conducted in the blank spaces. Furthermore, due to the independence of the Monte Carlo samples, the points lack consistency. The QMC method carefully designed a sequence of associated points to remove such clustering effects. Moreover, it also tries to prevent these phenomena and offer quicker convergence to the integral. Classic QMC sequences have a hierarchical framework: the integrand is sampled by the initial point on a coarse scale, and the following points sample it more finely.

Unofficially, the integration mistake related to a sequence relies on measuring the variation of the integrand f over the integration domain. Moreover, a sequence-dependent term typically measuring the discrepancy, or extent of departure from uniformity, of the sequence *S*. For instance, the expected Monte Carlo integration mistake decouples into a variance term and 1/*sqrt*(*s*) as in (9). A classic QMC theory formalises instinct as follows.

According to the Koksma-Hlawka inequality [35], for any function f with bounded variation, and a random set of points $S = W_1, \ldots, W_s$, the integration mistake is bounded above as it shows below:

$$\in s\left[f\right] \le D^*\left(S\right)V_{HK}\left[f\right] \tag{2.18}$$

in which $V[HK]$ is the variance of f in the meaning of Hardy and Krause defined by the following partial derivatives [36],

$$V_{HK}\left[f\right] = \sum_{I\in[d],I\neq 0}\int_{[0,1]^{|I|}}\left|\frac{\partial f}{\partial uI}\right|u_j = 1, j\notin I\left|duI\right., \tag{2.19}$$

where D^* is the star discrepancy, whose definition is:

$$D^*\left(S\right) = \sup_{X\in[0,1]^d}\left|disr_s\left(X\right)\right| \tag{2.20}$$

in which $disr_S$ is the local discrepancy function

$$disr_s\left(X\right) = Vol\left(J_x\right) - \frac{\left|\left\{i : W_i \in J_X\right\}\right|}{S} \tag{2.21}$$

with $J_X = [0, x_1) \times \cdots \times [0, x_d)$. The infinite sequence W_i mentioned above is defined to be a low-discrepancy sequence when the situation is as a function of s, the star discrepancy $D^*(\{W_1, \cdots, W_S\}) = O\left((\log S)^d / S\right)$.

Based on low-discrepancy sequences and the Koksma-Hlawka inequality, the classic Quasi-Monte Carlo method obtains a convergence rate of $O((\log S)^d / S)$. Although it is superior to $O(1/sqrt(S))$ for a stationary d, it needs s to be exponential in d to show the progress. Due to the factor above, the QMC method was rejected as unsuitable for high-dimensional integration.

2.5.2. Low-Discrepancy Sequence

Before this project introduces the low-discrepancy sequences, the definition of Discrepancy should be explained further. According to the definition of Discrepancy (2.15) [36], in simple words, the expression can be described as randomly selecting an area in space, the ratio of the number of points in this area to the total number of points minus the area of the space. Then the maximum number of the absolute value of their difference is Discrepancy. Next, before introducing these sequence definitions, this project will introduce a basic operation, Radical Inverse.

The radical inverse of an integer i with L digit base b expansion is defined as: $i = (i_{L-1}, i_{L-2}, \cdots, i_0)_b$.

And the expansion can be obtained via mirroring the numbers at the fractional point, $\varphi(i) = (0.i_0 i_1 \cdots i_{L-1})$.

This is also the definition of van der Corput sequence [37].

Digit Nets and Lattice Rules are two main kinds of methods for generating low-discrepancy sequences. For example, Halton sequence, Sobol sequence, Faure sequence and Niederreiter(t, s) sequence all belong to the kind of Digital Nets. The low-discrepancy sequence has the characteristic that for T's values, whose subsequence s_1, …, s_T has a low discrepancy. Besides, a low-discrepancy sequence is sometimes in the name of quasi-random sequences because these sequences are usually applied to replace random numbers.

The quasirandom sequences have different uses, and they have an advantage compared with pure random sequences, in which they quickly and uniformly cover the domain of interest. They also have an advantage over strictly deterministic approaches in that they only provide high accuracy when data points are pre-set. When employing quasirandom sequences, on the other hand, accuracy improves as additional data points are provided, with full reuse of the existing points. In contrast to totally random sequences, quasirandom point sets can have a substantially lower disparity for a given number of points. Finding the characteristic function of a probability density function is one of the practical uses. Finding the derivative function of a deterministic function with a small quantity of noise is the other one. Furthermore, higher-order moments may be calculated rapidly and accurately using quasirandom numbers.

3. Requirements and Analysis

In this chapter, this project will introduce the goals of this project in detail and

the analysis of the objectives. Then, this project will prepare some methods based on the review mentioned above. Finally, the evaluation of the result will be discussed.

3.1. Project Requirements

The project aims at implementing a new sampling scheme and applies the scheme into the kernel approximation based on the LFMs [3]. Therefore, this project will try to implement four kinds of low discrepancy sequences to obtain the random Fourier features for the approximation of the EQ kernel.

The requirements of the project are listed below:

1) Analyse the methods to obtain random Fourier features.

2) Implement the QMC sampling scheme.

3) Replace the MC approximation with the QMC approximation.

Besides these steps related to implementation, there are other requirements about this project. This project will carry out the new sampling schemes on different data sets and different models to fully verity the performance. Moreover, detailed experiment results and figures will be posted to make comparisons among all the sequences, thus the conclusion drawn from the project will be move reasonable and convincible.

3.2. Analysis

To satisfy the requirements of the project, this project will discuss the methods to meet the requirements mentioned above. The analysis of the each step will be analysed in the following subsections.

3.2.1. Random Fourier Features for EQ Kernel

According to the review about the LFMs, the expression for $k_q(\tau, \tau)$ is obtained. Moreover, based on the Bochner theorem [27], this project can transform the expression with a random Fourier feature expression.

$$k_q(T,T') = \mathrm{e}^{-\frac{(T,T')^2}{e_q^2}} = \int p(\lambda)\mathrm{e}^{j(T,T')\lambda}\mathrm{d}\lambda \tag{3.1}$$

in which $P(\lambda)$ obeys the distribution. Rahimi and Recht [6] gave a insight about approximating the $K_q(\tau, \tau^0)$ with the MC methods to solve the integral related to λ.

$$k_q(T,T') \approx \frac{1}{S}\sum_{S=1}^{S}\mathrm{e}^{j\lambda_s T} - \mathrm{e}^{j\lambda_s T'} = \frac{1}{S}\sum_{S=1}^{S} v(T,\lambda_S)v^*(T,\lambda_S) \tag{3.2}$$

in which S denotes the number of samples, and v is function with λ_s, meanwhile the function v^* is the complex conjugate of v, and $\lambda_s \sim p(\lambda)$. Due to the kernel function is a real number, we only need the real part of the inner product of the $v(\tau, \lambda_s)v^*(\tau, \lambda_s)$. Now we have the Equation (3.1), then we use replace the tradition $k_q(\tau, \tau^0)$ with the transformed expression for the $k_{fdfd0}(t, t^0)$ which is the cross-covariance of the LFM, we get

$$\sum_{q=1}^{Q} S_{d,q} S_{d',q} \int_0^t G_d(t-T) \int_0^{t'} G_{d'}(t'-T') \times \int p(\lambda) \mathrm{e}^{j(T-T')} \mathrm{d}\lambda \tag{3.3}$$

According to Alvarez [3], the above expression can be organized as

$$\sum_{q=1}^{Q} \frac{S_{d,q} S_{d',q}}{S} \left[\sum_{S=1}^{S} v_d(t, \theta_d, \lambda_s) v_d(t', \theta_{d'}, \lambda_s) \right], \tag{3.4}$$

in which $\lambda_s \sim p(\lambda)$, with

$$v_d(t, \theta_d, \lambda) = \int_0^t G_d(t-T) \mathrm{e}^{j\lambda_T} \mathrm{d}T \tag{3.5}$$

While, the expression of $v_d(t, \theta_d, \lambda)$ can still has a further extension. The θ_d refers to the Green's function's parameter $G_d(\cdot)$, and the integral over t can be solved using Laplace transform $L(\cdot)$.

$$\begin{aligned} v_d(t, \theta_d, \lambda) &= L^{-1} L\left\{ \int_0^t G_d(t-T) \mathrm{e}^{j\lambda_T} \mathrm{d}T \right\} \\ &= L^{-1}\left\{ g_d(s) L\left\{ \mathrm{e}^{j\lambda_T} \right\} \right\} \end{aligned} \tag{3.6}$$

in which $g_d(s)$ denote the Laplace transform for $G_d(t)$, and the function L^{-1} represents the inverse Laplace transform. Via the same steps, the expression is $v_d(t', \theta_{d'}, \lambda_s)$. Therefore, to obtain the approximation value of the LFM, there are two steps,

1) Calculate the expression for v_d using the Laplace transform.

2) Calculate the RFF approximation for the Equation (3.2), in which $\lambda_s \approx p(\lambda)$.

After analysing the steps above, there are two findings as listed below:

1) In step 1, v_d is a function related to Green's function, and the function $g_d(\cdot)$ changes according to different models. Besides, the the function is in symbol of the dynamic system's response to the excitation with referent to time, and it is defined as random Fourier response feature [3]. In step 2, the expression is approximated by the MC methods, in which the S denotes the number of samples. The project will focus on this part which is related to the kernel and the sampling method.

3.2.2. Implement the QMC Sampling Scheme

The QMC skill is usually applied when integrals are on the unit cube. Therefore, to realise the Quasi-MC method, a low-discrepancy sequences should be generated firstly. The points should be drawn regularly in a determined interval, $e_1, \cdots, e_s \in [0,1]^d$. Then the sequence should be transformed into a new sequence $w_1, \cdots, w_s \in R^d$, which is completely different from the methods used to generate the MC sequence. Define the density function in the Equation (2.8), in which $p_n(\cdot)$ denote a uni-variate density function. Moreover, the density function can make changes according to different types of shift-invariant kernels. Compared to the Avron's model [26], whose transformation is made on the Gaussian kernel. While, in this project, the kernel this project used is based on the LFMs, whose kernel is Exponentiated Quadratic kernel. To transform the equation 2.8 into an integral unit cube, making a simple change to the variable will be sufficient.

$$\Phi^{-1}(e)=\left(\Phi_1^{-1}(e_1),\cdots,\Phi_d^{-1}(e_d)\right)\in R^d, \tag{3.7}$$

in which function p_n has a cumulative distribution function Φ_n, for $n = 1, \ldots, d$. Via making $w = \Phi^{-1}(t)$, the Equation (2.8) can be modified as

$$\int_{R^d} \mathrm{e}^{j(T-T')w} p(w)\mathrm{d}w = \int_{[0,1]^d} \mathrm{e}^{j(T-T')\Phi^{-1}(e)}\mathrm{d}e \tag{3.8}$$

Therefore, this project can transform the sequence $e_1,\cdots,e_s\in[0,1]^d$ with function $w_i = \Phi^{-1}(e_i)$. Then, the transformed sequence can be applied in Equation (2.10) to obtain the QMC feature map. In conclusion, the QMC sampling scheme can be realised once the low discrepancy sequence is determined. This project has noticed the classic low discrepancy sequences, such as the Halton and Sobol sequences, which will be experimented with later. After the two simple experiments, more sequences will be experimented with to validate the improvement.

3.2.3. Replace the MC Approximation with QMC Approximation

According to the analysis of above parts, this project will explain the methods to replace the Sampling method as follow steps:

1) The shift-invariant kernel should be determined. The detailed definition can be found according to Bochner's theorem [27]. Here, the project uses the EQ kernel.

2) Decide p, which is the inverse Fourier transform of function k.

3) Choose a kind of low-discrepancy sequence to generate.

4) Make transformation to the sequence $e_1, \ldots, e_s$ with function $w_i = \Phi^{-1}(e_i)$ by Equation (3.3).

5) Obtain the feature map $\hat{\Psi}(x): R^d \to C^s$.

Notice that, according to the Section 3.2.1, when carrying out the project, the approximations are about v_d and v_d^*. This project will use the obtained RFRF for the approximations.

3.3. Evaluation

In this section, two aspects of evaluating the project will be analysed. Then, the detailed experiment results will be taken into consideration to compare their performance. Therefore, this project will design the evaluation metrics for the results.

From the aspect of meeting the requirements of the project this project will be evaluated according to the fulfilment of the sampling scheme, the replacement of the approximation, and the quality of new obtained random Fourier features.

From the aspect of assessing the performance of the LFMs. Evaluation metrics should be determined. The experiment results should be assessed according to the metrics. This project will make tables and figures to show the comparisons between all the sequences, and therefore, the conclusion drawn from the project is reasonable and convincible.

Evaluation Metrics

Normalised mean square error (NMSE), Negative log Predictive Density (NLPD) and the running time per iteration are three factor that account for the performance. The mean squared error method, which calculates the average squared error between the estimated and real values, is perhaps the most popular way to determine the loss. When the prediction has a small range, the denominator of NMSE could still work as penalty terms. Moreover, the model could predict every data example with the same value.

The normalised mean square error, also standardised mean square has the expressions as:

$$\text{NMSE} = \text{mean\_square\_error}/\text{mean\_tase\_variance} = \frac{\sum_i \left(y_i^* - y_i\right)^2}{\sum_i \left(\bar{y} - y_i\right)^2} \quad (3.9)$$

In expression, $\bar{y}$ denotes the mean value of y, y denotes the true output values, y^* denotes the predicted value of y.

Minimising the negative log predicted density is the same as maximising the log predicted density. People use the negative log predicted density (NLPD) as the loss function because they are used to minimising the loss function, and most work related to optimisation is done by minimising the result. The negative log predictive density (NLPD) is mainly used for a Gaussian predictive distribution to compute the negative log predictive of the test points. NLPD belongs to another type of performance measuring. It not only shows the accuracy of predictions but also shows the variance predictions of the posterior density. The expression of NLPD [38] is listed as below:

$$\text{NLPD} = \begin{bmatrix} \log \int \left(p\left(y_1|\theta\right)\right) p_{post}\left(\theta\right) \mathrm{d}\theta \\ \log \int \left(p\left(y_1|\theta\right)\right) p_{post}\left(\theta\right) \mathrm{d}\theta \\ \vdots \\ \log \int \left(p\left(y_1|\theta\right)\right) p_{post}\left(\theta\right) \mathrm{d}\theta \end{bmatrix} \quad (3.10)$$

where y_i denotes the predicted value of instance i, p_{post} denotes the posterior distribution.

In simple, to compute the NLPD in this project, three inputs are needed which are:

1) *ytrue* represents the true output values.

2) *ymu* represents the predictive means.

3) *yvar* represents the predictive variances.

Therefore, the expression of the output can be obtained as:

$$\text{NLPD} = \frac{1}{2}\frac{1}{T}\sum_{t=1}^{T}\left(\frac{\left(ytrye_t - ymut_t\right)^2}{yvar_t} + \log\left(2 * pi * yvar_t\right)\right). \quad (3.11)$$

To conclude, a model with smaller NMSE and NLPD performs better in discovering outputs.

4. Design

In this chapter, this project designs the developing process according to the analysis referred in Chapter 3. In the first section, this part illustrates the Latent Force model, and the detailed models to run the experiments. Then, the quasi-random sequences to implement the QMC method will be decided.

Finally, the project goes to the introduction of the developing platform.

4.1. Experiment Models and Data Sets

In this project, the first-order and second-order models described in Gao *et al.* [2] with the EQ kernel are approximated by the random Fourier response feature [3]. Besides, all the experiments are carried out with AMD ryzen 9 5900 HX @3.30 GHz. For the first order experiment, this project chooses the datasets about air temperature which include the measurements of air temperature at four different places which are Bramblemet, Cambermet, Sotonmet and Chimet [39]. The air temperatures from July 10 to July 15 in 2013 are measured for this project. The same experiment [3] [40] is adopted by this project, and the dataset is described in Table 1.

This project prefers to sufficiently assess the performance of the new sampling scheme, hence, the project carries out the experiments on the second-order model, which are more complex and computationally expensive. For the following experiments, this project considers the MOCAP datasets [41]. The datasets consist of measured actions, and one of the datasets is the movement of "Golf swing", which is stored in subject 64 motion 1. The data in this set are with a sudden curve and very changeable. The other movement is "Walk", which is stored in subject 02 motion 01. The data in this set are very smooth, which describe and record the movement of people's walk. With the changeable dataset and smooth dataset, this project is able to sufficiently evaluate the performance of the random Fourier response features via the new sampling scheme.

4.2. Experimental Sequences

In this project, four kinds of low-discrepancy sequences are designed for approximations. The parameters and details are listed as follows. Notice that the radical inverse method has been introduced in Chapter 2.

Table 1. Detailed number of data points used in the experiment.

#	Name	Training	Test
1	Bramblemet	1425	0
2	Cambermet	1268	173
3	Chimet	1235	201
4	Sotonmet	1097	0

4.2.1. Halton Sequence

Halton sequences are mainly used in statistics, which are used to generate point sets for different sampling methods such as QMC and MC simulations. While these sequences are deterministic, their discrepancy are low.

The definition of Halton sequence is: $X_i := \left(\varphi_{b_1}, \cdots, \varphi\left(b_n\right)\right)$, where b_i and b_i are coprime numbers. For example, this project will take one dimension sequence based on 2 and 3. To generate the points, the interval (0, 1) will be divided in half, then in fourths, etc., leading to the points:

$$\frac{1}{2}, \frac{1}{4}, \frac{3}{4}, \frac{1}{8}, \frac{5}{8}, \frac{3}{8}, \frac{7}{8}, \frac{1}{16}, \cdots$$

To generate the sequence based on 3, the interval will be divided in thirds, then ninths, etc., leading to the points:

$$\frac{1}{3}, \frac{2}{3}, \frac{1}{9}, \frac{4}{9}, \frac{7}{9}, \frac{2}{9}, \frac{5}{9}, \frac{8}{9}, \cdots$$

After the two points sets are generated, this project can get a sequence in a unit cube by pairing the above two point sets:

$$\left(\frac{1}{2}, \frac{1}{3}\right), \left(\frac{1}{4}, \frac{2}{3}\right), \left(\frac{3}{4}, \frac{1}{9}\right), \left(\frac{1}{8}, \frac{4}{9}\right), \left(\frac{5}{8}, \frac{7}{9}\right), \left(\frac{3}{8}, \frac{4}{2}\right), \left(\frac{7}{8}, \frac{5}{9}\right), \left(\frac{1}{16}, \frac{8}{9}\right)$$

While the standard Halton sequences do well in low dimensions, there are correlation problems when generating the sequences with high primes. One way to avoid correlation problems is to scramble Halton sequence. The function used to generate Halton sequences is haltonset(). The properties of the function are shown as follows:

1) Dimensions means the number of dimensions. For example, the number 5 represents a creationof five-dimensional point set. When using the haltonset() function to create the point set, the input argument decide the number of dimensions.

2) Leap means the interval between points. The value of this input parameter is specified as a positivenumber. The property of the leap parameter decides the number of points that will be omitted and leapt over. Usually, the default value is 0, which means this sequence takes its every point. To improve the quality of point sets, the leaping skill can be used. While, when choosing the leap values, patience should be paid on. Many leap values fail to be uniform quasirandom point sets because the sequences they create can not reach the sub-hyper-rectangles of the unit hyper cube [42]. There is a rule about choosing the leap values, as n is a prime number which is not be in use to generate the dimensions, then (n − 1) can be set to be the leap value. For instance, for a point set with d-dimensional, determine the $(d+1)^{th}$ or greater primer number for n.

1) Skip means the number of points will be omitted at the beginning.

2) Scramble. The parameter controls the scrambling of the sequence, and it specifies the type and option of the scrambles.

Hence, in this project, we will generate the points by the function.

4.2.2. Sobol Sequence

The motivation for creating the Sobol sequence is to construct a sequence with a fast rate of convergence. Let $R^D = [0, 1]^D$ be a d-dimensional unite hypercube, and f is a integral function in the interval R^D. A sequence x_n in R^d is applied in the expression showed below,

$$\lim_{n\to\infty}\frac{1}{N}\sum_{n=1}^{N} f\left(x_n\right) = \int_{R^D} f \tag{4.1}$$

To make the sum has a convergence towards the integral, R^D should be filled by the points x_n to minimise the holes. The x_n has a good property that its projections leave few holes on a low-dimensional space of R^D.

Different with Halton sequence, all the dimensions of Sobol sequence are consists of radical inversion with base 2, but each dimension has a different generating matrix C. Hence, $X_i := \left(\varphi_{2,C_1}(i), \cdots, \varphi_{2,C_n}(i)\right)$.

Similar to the Halton sequence, the properties of the parameters are listed below:

1) Dimensions specify the number of dimensions.

2) Leap denotes the interval between points.

3) PointOrder specifies the rule of generating points. Usually, the default setting is "graycode".

4) Scramble. The parameter controls the scrambling of the sequence, and it specifies the type andoption of the scrambles.

5) Skip denotes the number of initial points to be omitted.

In this project, the default settings are $skip = 1000$ and $leap = 700$.

Considering the Sobol sequence need a generating matrix, this project refers to a tool [43] to obtain the generating matrix.

4.2.3. Digital Nets

Digital sequences are generated by a set of generating matrices rather than generated by an integer generating vector as is the case for lattice sequences. This project takes them as a vector of matrices which are with elements from a finite field. For instance, if a digital sequence is in base 2, then its finite field is simply Z2.

Similar to the function introduce above, the function used to create Digital Nets sequences is $p = digitalseq_b2g(d, s)$. The function means the points are generated from the base 2 digit sequence in the reverse order of the grey-coded root. The input parameters are listed below:

1) d denotes the number of dimensions per vector, 2) s denotes the number of samples.

The output is x which denotes an array of sample vectors.

The usages to generate the sequences are as followed:

1) A point set with generating matrices should be prepared. The point set this project uses is generated by public available tools, and the sequence to generate the points is the "new-joe-kuo-6.21201" [44]. The detailed information for gen-

eration points will be put on the appendix.

2) The generator should be initialised first for a point set. There are three options to choose: *init*0 means the sequence keep as it is *init*1 means the first point *initskip* means the first point in the sequences is omitted

3) Generate the next $sd - vetors$ of the sequence, returning a d-by-s dimensional array.

$p = digitalseq_b2g(d, s)$ 4. Scramble. The parameter controls the scrambling of the sequence, and it specifies the type and option of the scrambles.

4.2.4. Lattice Rules

Similar with the Sobol sequence. The Lattice rules need a generating vector Z to build the sequences.

According to the radical inverse introduced in Chapter 2, the sequence is given as: $X_i = \varphi_b(i)Zmod\,[0, 1)^n$. In simple words, if the $\varphi_b(i)Z$ is larger than 1, then the modulate of 1 is mapped back to the range of [0, 1).

The properties of parameters are listed below:

1) d denotes the number of dimensions per vector.

2) s denotes the number of samples.

The usages are also listed below:

1) The generator should be initialised first, and there are three valid options: "init0", "init1", and "initskip". The definitions of these three parameters are the same with Digit Nets.

2) Generate the sequences, and the returned value is a d-by-s dimensional array. In this project, the default parameter is "initskip", $latticeseq_{b2}({}^{0}initskip^{0})$

4.3. Experiments Design

The design is straightforward. After implementing four kinds of low discrepancy sequences, they will be tested with three data sets for a first-order and a second-order model to fully evaluate their performance in different situations. More detailed information about implementations is in Chapter 5.

4.4. Library and Environment

The whole project is built by MATLAB. The additional toolbox which is Statistics and Machine Learning Toolbox is required in this project. Besides, this project uses some tools in MATLAB and public implementation [43].

5. Implementation

In this chapter, this project introduces some codes related parts. The project plans to use four sequences: Lattice Rules, Halton, Sobol, and Digit Nets. For the two simple sequences, Halton and Sobol sequence, the project will use the implementation in MATLAB. While for Lattice Rules and Digit Nets, the scrambling and shifting technologies the project will use are adapted from QMC literature review [45].

5.1. Preprocessing Data

This project uses the Air experiment as an example. There are two steps to preprocess the extracted data.

1) Load data: *load../datasets/weather/weatherdata.mat.*

2) This project split the training and testing data. The project will the data about Chimet and Cambermet where the testing set is chosen according to the time, and the remains are training set.

test ind1 = xT{2} >= 10.2&xT{2} <= 10.8;

test ind2 = xT{3} >= 13.5&xT{3} <= 14.2;

outs = [2, 3].

5.2. Generating the Sequences

According to the design in Chapter 4, this project will generate the sequence with the designed parameter. For example, the Halton sequence is generated with the Matlab tools, *p* = *Haltonset*((*d*, "*Skip*", *skip*, "*Leap*", *leap*);) and scrambling technologies is applied for it: (*d*, "*Skip*", *skip*, "*Leap*", *leap*);. Then, this project can get *points* = *p*(1: *s*, 1: *d*);. Finally, the points should be transformed: *W* = *norminv*(*points*, 0, 1).

Besides, the generator of Lattice rules and Digital nets are different from the Sobol sequence. Lattice rules need a generating vector, and Digital nets need a generating matrix. According to the literature [43], this project uses an extra tool called MinGW in Windows. The steps for obtaining the points are listed below:

g + + − *osobolsobol.cc*,

./sobol 10 3*new* − *jok* − *kuo*, choose the size of the points.

Then, the steps are similar to the Halton sequence.

5.3. Options Setting

After processing the extracted data, the next step is to set the kernel options. For example, some important parameters are listed below:

1) *kern.Type* = "*kffsim*". This project will selesct different kernels according to the datasets.

2) *model* = *dtcmgpOptimise*(*model*, 2,500). This project will choose a different number of iterations to explore the rate of convergence. Here, 500 is the number of iteration times.

3) *options.kern.S* = 100. This project will decide the number of sampling points. Usually, the results get more accurate as the number of sampling points is bigger.

Besides, it is necessary to select the sequence and sampling points. There are five different sampling points that can be generated in this project: "MC", "Halton", "Sobol", "Lattice rules", and "Digital nets". In the file "*kernParamInt*", this project chooses the sequences for experiments. For example, *sequence* = "*Halton*". Once the sequences are determined, the scrambling skills are applied.

6. Results and Discussion

In this section, this project carries out experiments based on two models and three datasets to sufficiently evaluate the performance of RFRF [3].

6.1. Experiment Results

In this section, this project mainly shows the results by two methods. One of them is to show the Gaussian Processes (GPs) to show the figure about fitting data directly. The other method is to record the results in tables for further comparisons.

As it is shown from the **Figure 3**, testing data are using blue dots, and training data are using pink dots. The black line in the figure represents the mean of the GP function, and the standard deviation is represented via the shaded areas.

6.1.1. First-Order Model with Air Dataset

In this model, this project will use the data set about air temperature 4.1. Five tables and their predictive GPs are listed below.

The results of air temperature are shown in **Table 2** obtained by Halton sequences. As the data shows above, when the sampling point is 10, the Halton sequence has already got an excellent result in the Cambermet dataset. On the other hand, when the project picked 50 sampling points, it has reached the lowest NMSE and NLPD in Chimet output (See **Figure 2**).

The results of air temperature are shown in **Table 3** obtained by the MC sampling method. As the data shows above, when the sampling points are 100, the MC sampling got an excellent result in Cambermet and Chimet datasets.

Table 2. Results with Halton sequence.

First-order model (Air)		Cambermet		Chimet		Time
Sequence	Samples	NMSE	NLPD	NMSE	NLPD	[s]
Halton+	S10	**0.09**	**1.06**	0.31	1.22	1.13
Halton+	S20	0.13	1.26	0.30	1.19	1.30
Halton+	S50	0.09	1.14	**0.23**	**1.07**	1.31
Halton+	S100	0.10	1.17	0.32	1.16	1.76

Table 3. Results with Monte Carlo method.

First-order model (Air)		Cambermet		Chimet		Time
Sequence	Samples	NMSE	NLPD	NMSE	NLPD	[s]
MC+	S10	0.20	1.53	0.85	1.66	1.16
MC+	S20	0.21	1.45	2.46	3.33	1.29
MC+	S50	0.16	1.36	0.62	1.51	1.42
MC+	S100	**0.12**	**1.18**	**0.28**	**1.1**	1.79

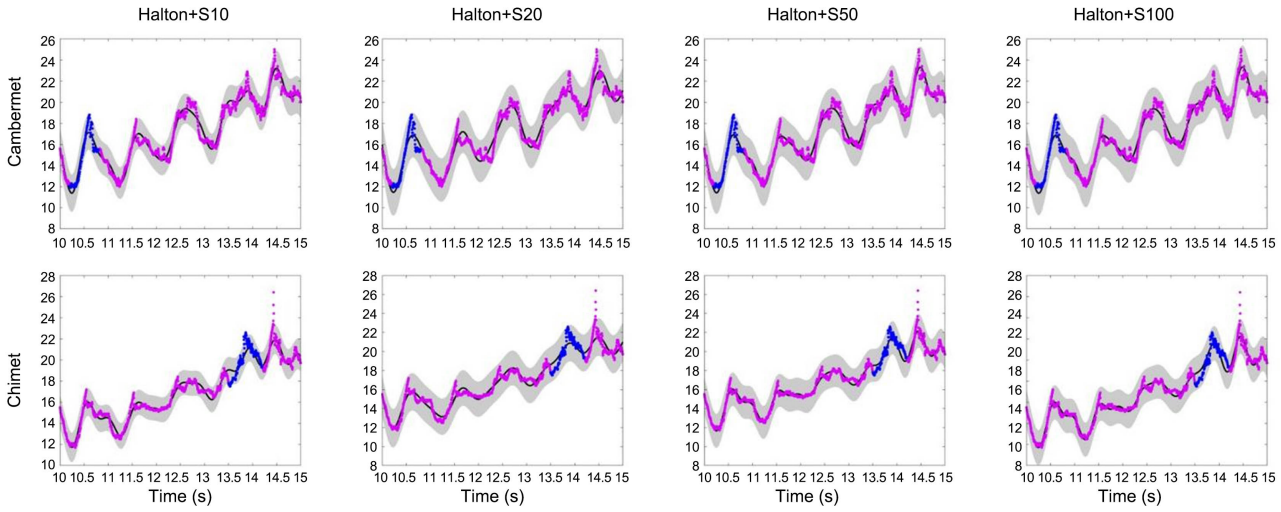

Figure 2. Predictive GPs with Halton sequence for the Air data set.

The results of air temperature are shown in **Table 4** obtained by Sobol sequence. As the data shows above, when the sampling points are 50, the Sobol sequence got an excellent result in Cambermet and Chimet data set (See **Figure 3**).

The results of air temperature are shown in **Table 5** obtained by Lattice rules. As the data shows above, when the sampling points are 100, the Lattice rules got an excellent result in the Cambermet dataset, and when the project picked 50 sampling points, the result of Chimet reached its lowest value (See **Figure 4**). Digital

The results of air temperature are shown in **Table 6** obtained by Digital nets. As the data shows above, when the sampling points are 100, the Digital nets got a good result in the Cambermet dataset, and the NLPD in Chimet reached its lowest value (See **Figure 5**).

Compare the five tables, we have noticed that the best result of MC sampling is 0.12 and 1.18 in the Cambermet dataset when the project uses 100 sampling points. While for other low-discrepancy sequences, the Halton sequence got 0.09 and 1.06 in Cambermet at a very early stage where the number of sampling points is 10. The Halton set performs much better than the traditional MC method. The table contains the best performance of each sequence in Cambermet is listed below (See **Figure 6**).

As **Table 7** shows, all the four low-discrepancy sequences have a better performance compared to the MC method, where the NMSE and NLPD of the QMC method are smaller than that in the MC method. Besides, the Halton sequence has already reached its relative best performance as ten sampling points, and the Sobol sequence also has excellent NMSE and NLPD at 50 sampling points. Moreover, from the tables, the project has found that the running time per iteration of each sequence is very similar when the sequences have the same number of sampling points. Avron *et al.* (2016) [26] has pointed out these low-

discrepancy sequences are generated very fast. Therefore, in the experiments of the second-order model, this project will not record the running time per iteration for comparison.

Table 4. Results with Sobol sequence.

First-order model (Air)		Cambermet		Chimet		Time
Sequence	Samples	NMSE	NLPD	NMSE	NLPD	[s]
Sobol+	S10	0.83	2.17	0.85	1.68	1.13
Sobol+	S20	0.23	1.65	0.81	1.70	1.23
Sobol+	S50	**0.08**	**1.00**	**0.37**	**1.33**	1.35
Sobol+	S100	0.16	1.49	0.75	1.61	1.80

Table 5. Results with Lattice rules.

First-order model (Air)		Cambermet		Chimet		Time
Sequence	Samples	NMSE	NLPD	NMSE	NLPD	[s]
Lattice rules+	S10	0.22	1.56	1.67	2.06	1.26
Lattice rules+	S20	0.25	1.61	0.84	1.65	1.30
Lattice rules+	S50	0.14	1.45	**0.24**	**1.30**	1.39
Lattice rules+	S100	**0.10**	**1.14**	0.65	1.57	2.12

Table 6. Results with Digital nets.

First-order model (Air)		Cambermet		Chimet		Time
Sequence	Samples	NMSE	NLPD	NMSE	NLPD	[s]
Digital nets+	S10	0.66	2.03	0.78	1.66	1.14
Digital nets+	S20	0.28	1.69	0.67	1.62	1.04
Digital nets+	S50	0.15	1.43	**0.22**	1.37	1.29
Digital nets+	S100	**0.12**	**1.17**	0.37	**1.25**	1.9

Table 7. Best results for each sequence for output Cambermet.

First-order model (Air)		Cambermet	
Sequence	Samples	NMSE	NLPD
Halton+	S10	0.09	1.06
Sobol+	S50	0.08	1.00
Lattice rules+	S100	0.10	1.14
Digital nets+	S100	0.12	1.17
MC+	S100	0.12	1.18

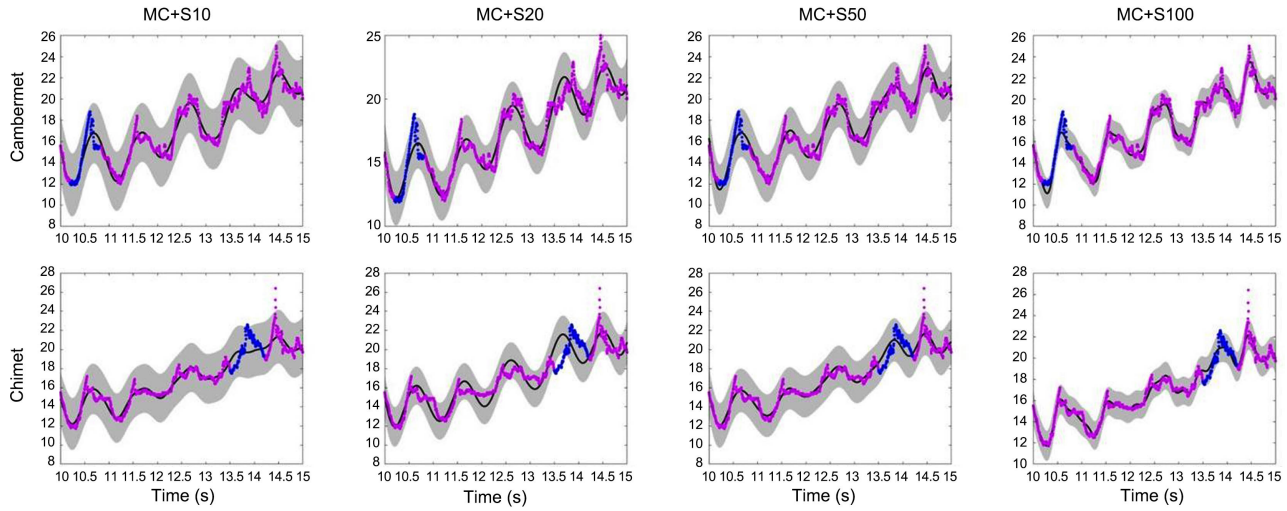

Figure 3. Predictive GPs with the MC method for the Air data set.

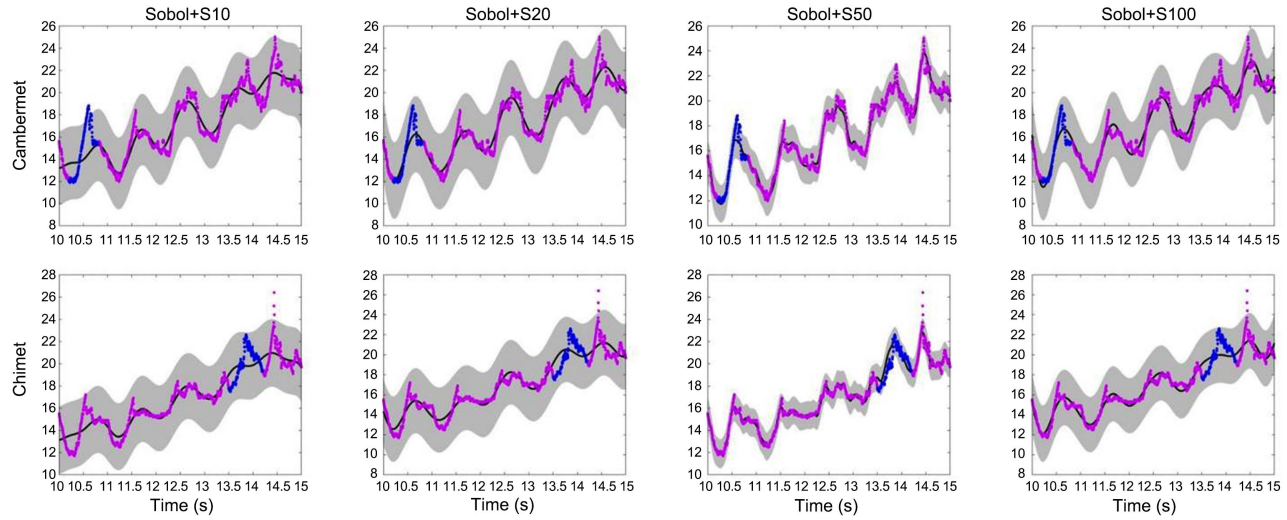

Figure 4. Predictive GPs with Sobol sequence for the Air data set.

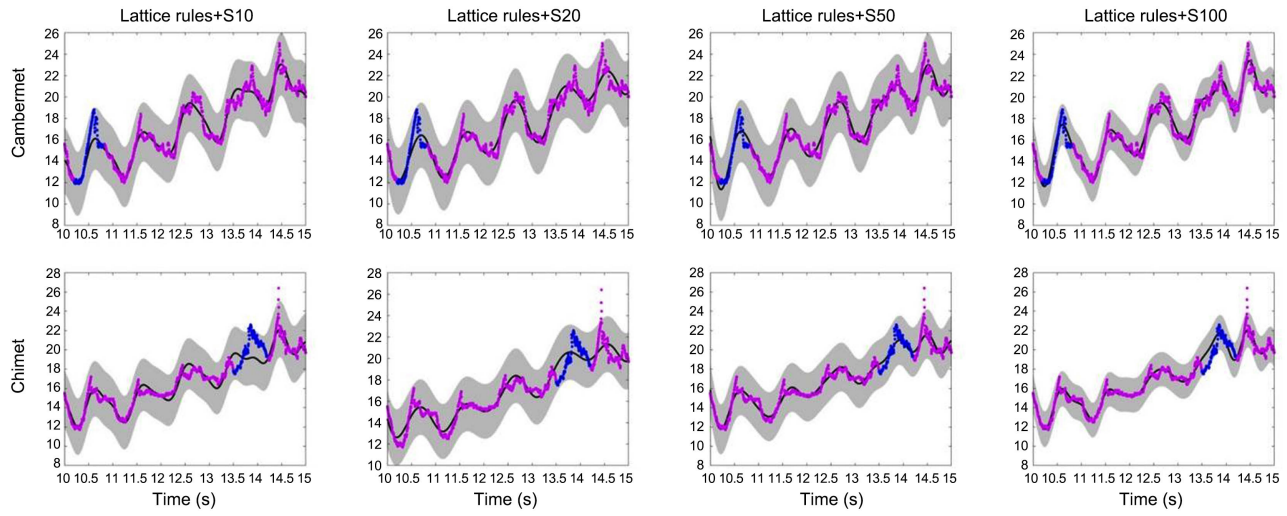

Figure 5. Predictive GPs with Lattice rules for the air data set.

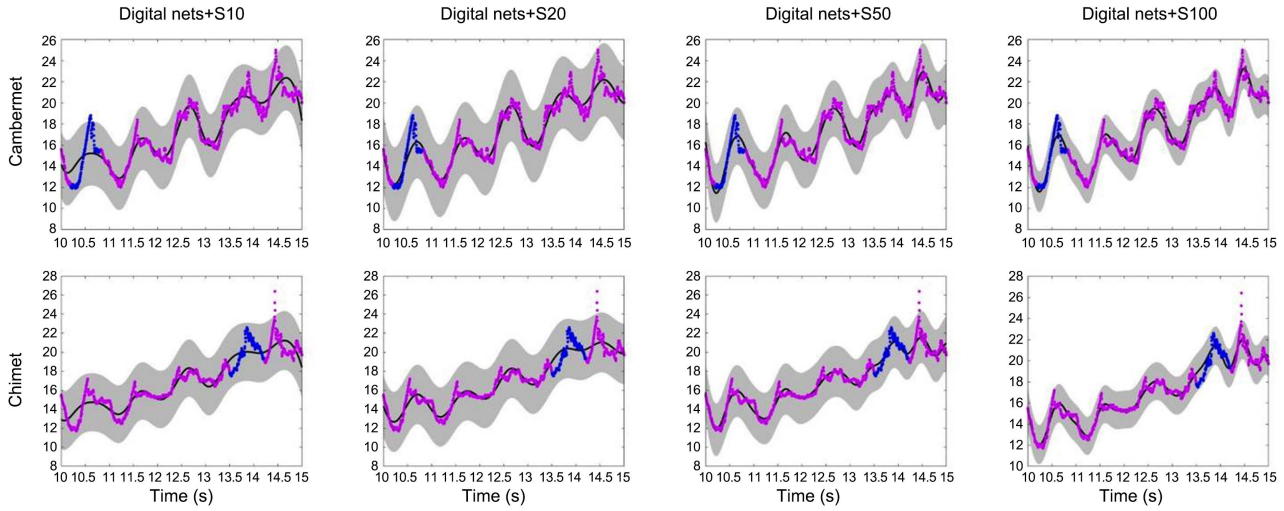

Figure 6. Predictive GPs with Digital nets for the Air data set.

6.1.2. Second-Order Model with Swing Dataset

In this model, this project will use the data set about Golf Swing movements, and it is carried out with the second-order model. Five tables and their GPs are listed below.

The results of Swing movements are shown in **Table 8** obtained by the Monte Carlo method. As the data shows above, when the sampling point is sixty, the Monte Carlo method gets a good result in both the root-Ypos and lowerback-Yrot outputs.

The results of Swing movements are shown in **Table 9** obtained by Lattice rules. As the data shows above, when the sampling point is sixty, the Lattice rules gets a good result in the root-Ypos output, and when the sampling points are 100, the lowerback-Yrot has a good result. Notice that when the sampling points are 60, the Lattice rules have already reached a relatively good performance (See **Figure 7**).

The Digital nets obtain the results of Swing movements in **Table 10**. As the data shows above, when the sampling point is sixty, the Digital nets got a good result in the root-Ypos dataset, and when the sampling point is 100, the sequence has its best results. Besides, we can notice that the Digital nets sequence has reached its relative best performance when the number of sampling points is 20 (See **Figure 8**).

The Sobol sequence obtains the results of Swing movements in **Table 11**. As the data shows above, when the sampling point is twenty, the Sobol sequence got a good result in the root-Ypos dataset, and NMSE in lowerback-Yrot has also minimised to lowest. Besides, when the sampling point is ten, the sequence has its best results in NLPD of the lowerback-Yrot output. Thus, in this experiment, the datasets of the Golf swing can be fitted by the proposed RFRF with few sampling points and enough accuracy (See **Figure 8**).

The results of Swing movements are shown in **Table 12** obtained by the Halton sequence. As the data shows above, when the sampling point is sixty, the

Halton sequence got a good result in the root-Ypos dataset. Besides, when the sampling points are 100, the sequence has its best results in lowerback-Yrot output (See Figure 9).

Table 8. Results with Monte Carlo method.

Second-order model (Swing)		root-Ypos		lowerback-Yrot	
Sequence	Samples	NMSE	NLPD	NMSE	NLPD
MC+	S10	0.49	−2.13	1.74	3.65
MC+	S20	0.17	−2.33	1.39	4.37
MC+	S60	**0.14**	**−2.39**	**0.93**	**2.6**
MC+	S100	0.22	−2.34	1.23	3.28

Table 9. Results with Lattice rules.

Second-order model (Swing)		root-Ypos		lowerback-Yrot	
Sequence	Samples	NMSE	NLPD	NMSE	NLPD
Lattice rules+	S10	0.45	−2.17	1.92	4.74
Lattice rules+	S20	2.04	−0.86	2.24	5.14
Lattice rules+	S60	**0.20**	**−2.38**	0.24	1.18
Lattice rules+	S100	0.28	−2.3	**0.14**	**0.81**

Table 10. Results with Digital nets.

Second-order model (Swing)		root-Ypos		lowerback-Yrot	
Sequence	Samples	NMSE	NLPD	NMSE	NLPD
Digital nets+	S10	0.49	-2.13	4.04	6.88
Digital nets+	S20	0.13	-2.41	2.33	4.87
Digital nets+	S60	**0.13**	**-2.42**	0.85	2.75
Digital nets+	S100	0.17	-2.40	**0.39**	**1.48**

Table 11. Results with Sobol sequence.

Second-order model (Swing)		root-Ypos		lowerback-Yrot	
Sequence	Samples	NMSE	NLPD	NMSE	NLPD
Sobol+	S10	0.34	−2.27	1.69	**3.00**
Sobol+	S20	**0.12**	**−2.43**	**1.24**	3.24
Sobol+	S60	0.12	−2.40	1.4	4.05
Sobol+	S100	0.28	−2.31	1.48	3.38

Table 12. Results with Halton sequence.

Second-order model (Swing)		root-Ypos		lowerback-Yrot	
Sequence	Samples	NMSE	NLPD	NMSE	NLPD
Halton+	S10	0.21	−2.36	2.16	3.94
Halton+	S20	0.51	−2.12	5.79	8.93
Halton+	S60	**0.12**	**−2.42**	0.17	0.93
Halton+	S100	0.17	−2.37	**0.12**	**0.79**

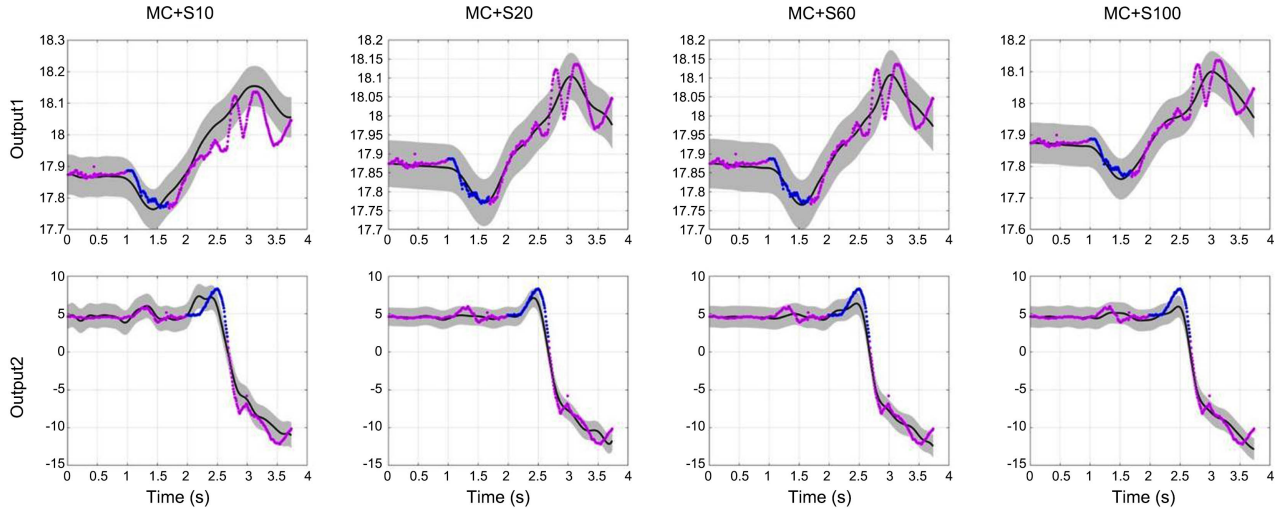

Figure 7. Predictive GPs with the MC method for the Swing data set.

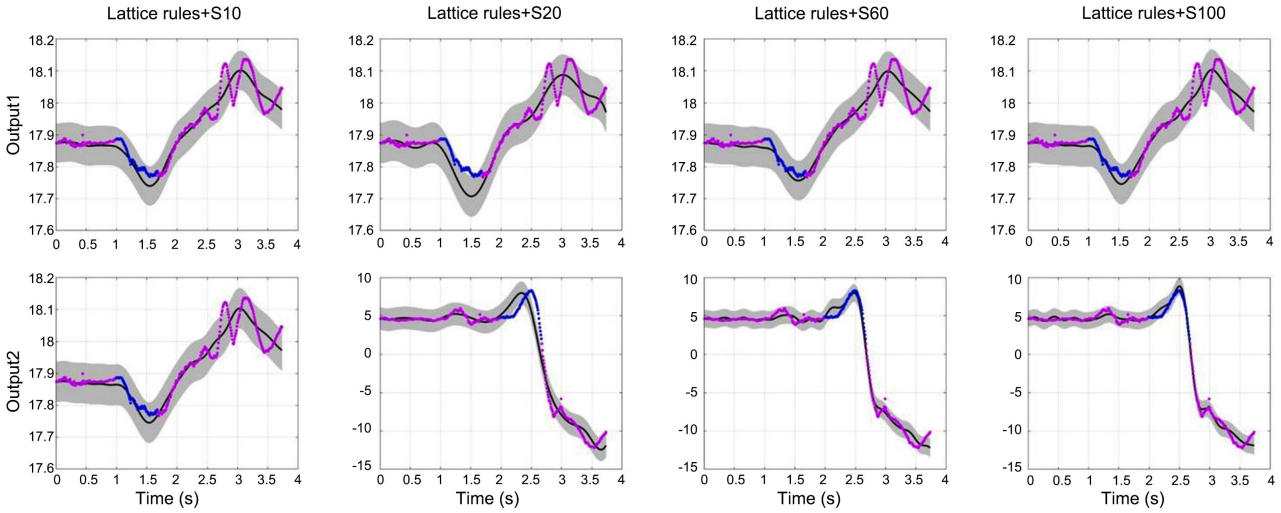

Figure 8. Predictive GPs with Lattice rules for the Swing data set.

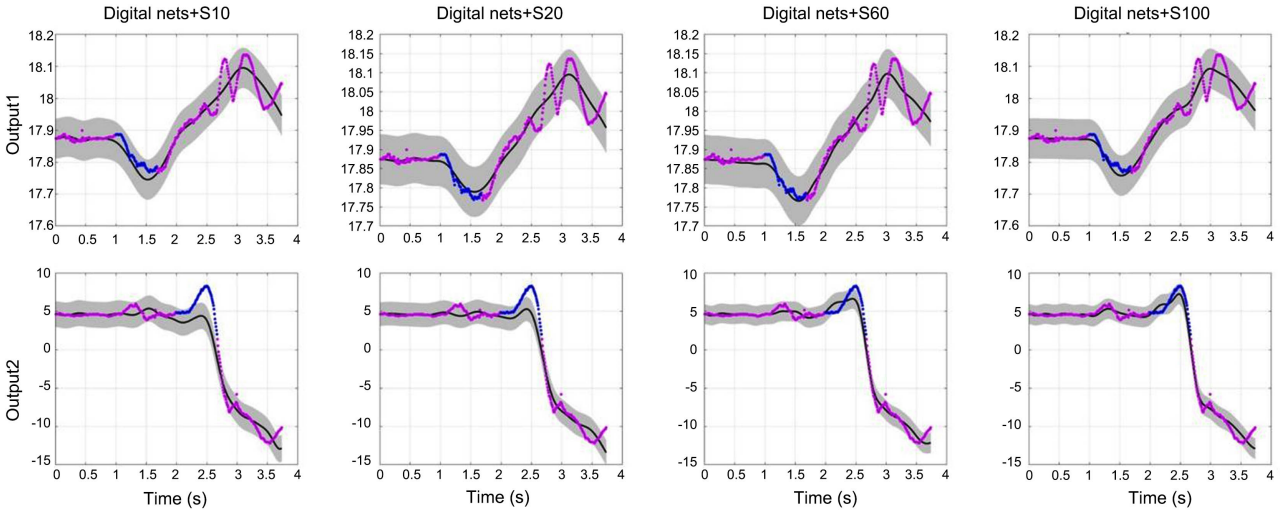

Figure 9. Predictive GPs with Digital nets for the Swing data set.

As the results are shown on Table 13, this project takes the output root-Ypos as an instance. Except for the Lattice rules, the Halton sequence, the Sobol sequence and the Digital nets all have smaller NMSE and NLPD. Besides, the results with 20 sampling points of the Sobol sequence and Digital nets are better than the MC method with 100 sampling points (See Figure 10).

6.1.3. Second-Order Model with Walk Dataset

In this model, the project will use the data set about Walk movements, which is carried out with the second-order model. Five tables of each sequence and their predictive GPs are listed below.

The results of Swing movements are shown in Table 14 obtained by the Halton sequence. As the data shows above, when the sampling point is ten, the Halton sequence got its best performance in lowerback-Yrot. Besides, when the sampling points are 30, the sequence has its best results in Iradius-Xrot output. So in this situation, the Halton sequence can fit the data well at a stage with few sampling points. Moreover, the result in Iradius-Xrot is excellent (See Figures 11-13).

Table 13. Best results for each sequence for Swing dataset output root-Ypos.

Second-order model (Swing)		root-Ypos	
Sequence	Samples	NMSE	NLPD
Halton+	S60	0.12	−2.42
Sobol+	S20	0.12	−2.43
Lattice rules+	S60	0.20	−2.38
Digital nets+	S60	0.13	−2.43
MC+	S60	0.14	−2.39

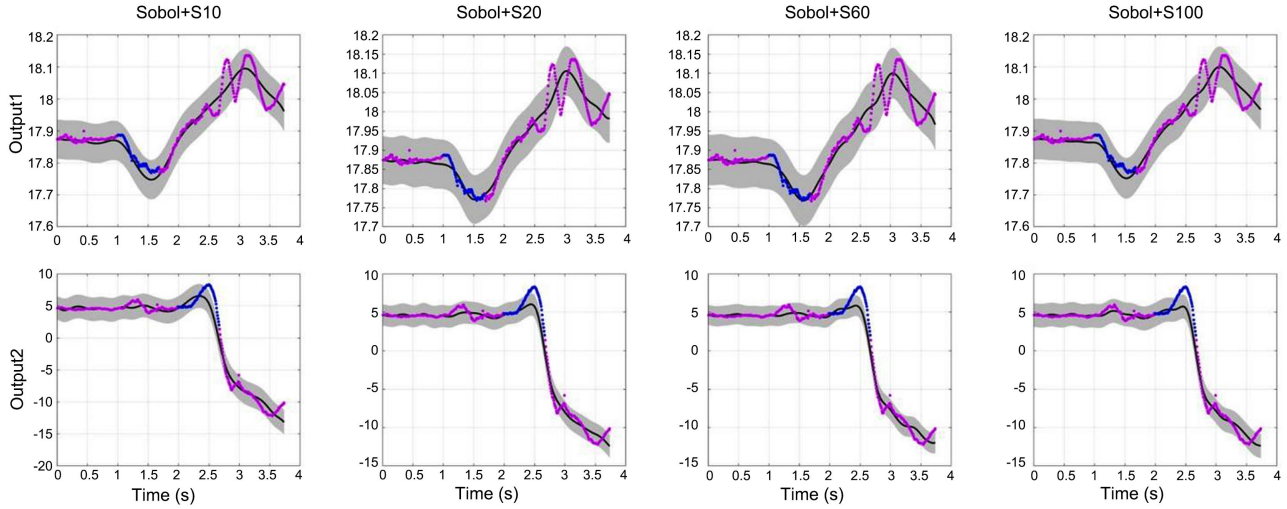

Figure 10. Predictive GPs with Sobol sequence for the Swing data set.

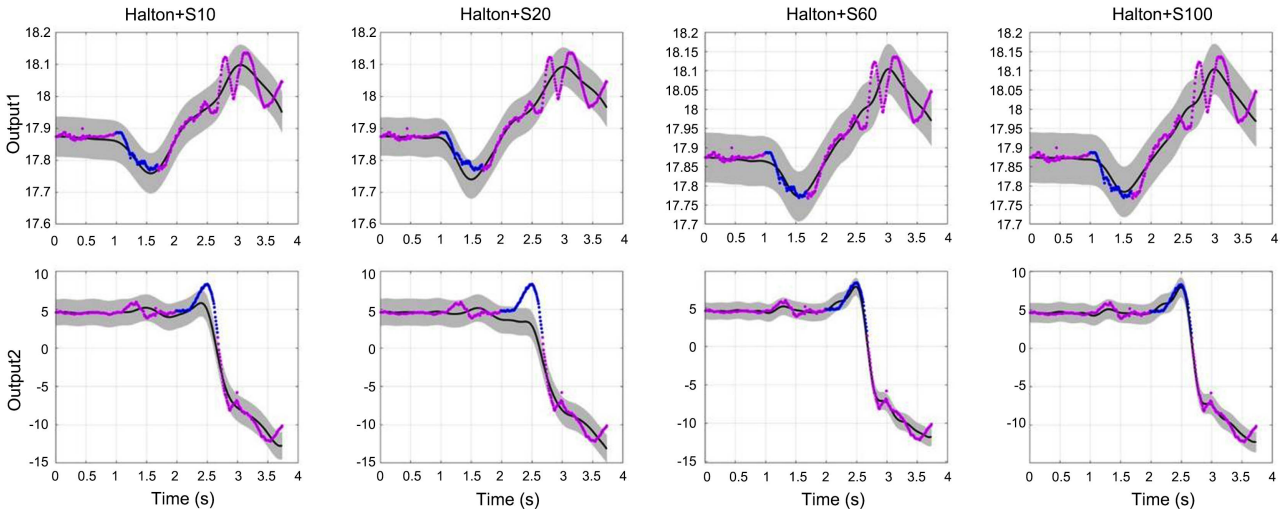

Figure 11. Predictive GPs with Halton sequence for the Swing data set.

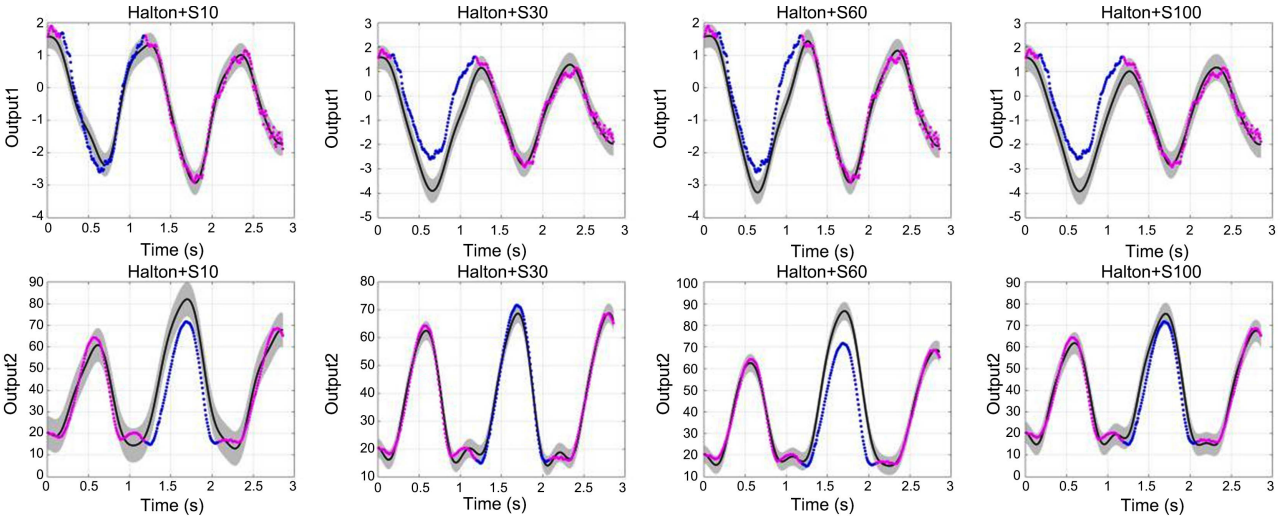

Figure 12. Predictive GPs with Halton sequence for walk data set.

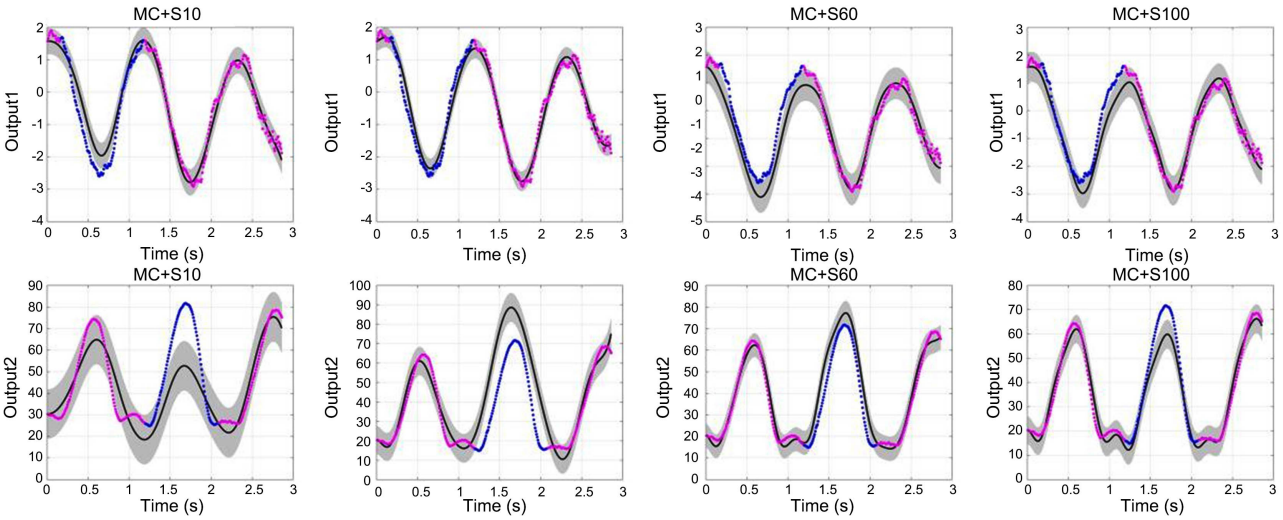

Figure 13. Predictive GPs with the MC method for walk data set.

Table 14. Results with Halton sequence.

Second-order model (Walk)		lowerback-Yrot		Iradius—Xrot	
Sequence	Samples	NMSE	NLPD	NMSE	NLPD
Halton+	S10	**0.05**	**0.93**	0.47	8.99
Halton+	S30	0.66	11.08	**0.01**	**2.25**
Halton+	S60	0.24	7.17	2.51	27.17
Halton+	S100	0.67	9.21	0.11	5.56

The results of Swing movements are shown in Table 15 obtained by the MC method. As the data shows above, when the sampling point is thirty, the Monte Carlo method got its best performance in lowerback-Yrot. Besides, when the sampling points are 100, the sequence has its best results in Iradius-Xrot output.

The results of Swing movements are shown in Table 16 obtained by the Lattice rules. As the data shows above, when the sampling point is thirty, the Monte Carlo method got its best performance in lowerback-Yrot. Besides, when the sampling points are 100, the sequence has its best results in Iradius-Xrot output.

The results of Swing movements are shown in Table 17 obtained by the Sobol sequence. As the data shows above, when the sampling point is thirty, the Sobol sequence got its best performance in lowerback-Yrot. Besides, when the sampling points are 100, the sequence has its best results in Iradius-Xrot output (See Figures 14-16).

The results of Swing movements are shown in Table 18 obtained by the Digital nets. As the data shows above, when the sampling point is a hundred, the Digital nets get their best performance in lowerback-Yrot and Iradius-Xrot. Besides, this project notices that when the sampling points are 30, the Digital nets can already have a relatively better performance than the MC method.

Then, this project takes the results of output lowerback-Yrot as an instance. As the table shows the results of output lower-Yrot (Table 19), the Mont Carlo method ranks 4^{th} in NMSE and ranks 5^{th} in NLPD among the results of output lower-Yrot. Interestingly, the Halton sequence is the fastest to reach its best performance with ten sampling points. Although its NMSE is a little bigger than the MC, its NLPD is smaller, so in this experiment, the Halton sequence's performance is similar to the MC method. For other sequences, their performance is better than the one of the MC method.

6.1.4. Discussion

This project has noticed that sometimes as the number of sampling points rises, the NMSE and NLPD get bigger. This situation is because this project is a machine learning project. When the complexity of the model is higher than the actual problems, the model learns the characteristics of the training set that is not suitable to the test set. For example, in the experiment of Air data set for the first-order model, most of the sequences need 100 sampling points to get a good performance in fitting the data.

Table 15. Results with MC method.

Second-order model (Walk)		lowerback-Yrot		Iradius—Xrot	
Sequence	Samples	NMSE	NLPD	NMSE	NLPD
MC+	S10	0.12	2.43	0.69	7.29
MC+	S30	**0.04**	**1.09**	0.98	17.95
MC+	S60	0.21	2.43	0.16	6.43
MC+	S100	0.12	1.42	**0.11**	**4.82**

Table 16. Results with Lattice rules.

Second-order model (Walk)		lowerback-Yrot		Iradius—Xrot	
Sequence	Samples	NMSE	NLPD	NMSE	NLPD
Lattice rules+	S10	0.25	3.66	0.94	18.61
Lattice rules+	S30	**0.03**	**0.67**	1.13	15.29
Lattice rules+	S60	0.50	9.02	0.69	29.19
Lattice rules+	S100	0.37	4.00	**0.45**	**10.56**

Table 17. Results with Sobol sequence.

Second-order model (Walk)		lowerback-Yrot		Iradius—Xrot	
Sequence	Samples	NMSE	NLPD	NMSE	NLPD
Sobol+	S10	0.21	3.57	0.29	14.40
Sobol+	S30	**0.02**	**-0.28**	0.21	3.86
Sobol+	S60	0.13	1.95	0.67	15.59
Sobol+	S100	0.13	3.49	**0.08**	**3.37**

Table 18. Results with Digital nets.

Second-order model (Walk)		lowerback-Yrot		Iradius—Xrot	
Sequence	Samples	NMSE	NLPD	NMSE	NLPD
Digital nets+	S10	0.53	9.74	2.15	30.47
Digital nets+	S30	0.03	0.70	1.31	18.33
Digital nets+	S60	0.37	6.39	0.38	10.99
Digital nets+	S100	**0.02**	**0.14**	**0.17**	**4.13**

Table 19. Best results for each sequence for Swing dataset output root Ypos.

Second-order model (Swing)		lower-Yrot	
Sequence	Samples	NMSE	NLPD
Halton+	S10	0.05	0.93
Sobol+	S30	0.02	−0.28
Lattice rules+	S30	0.03	0.67
Digital nets+	S60	0.02	0.14
MC+	S30	0.04	1.09

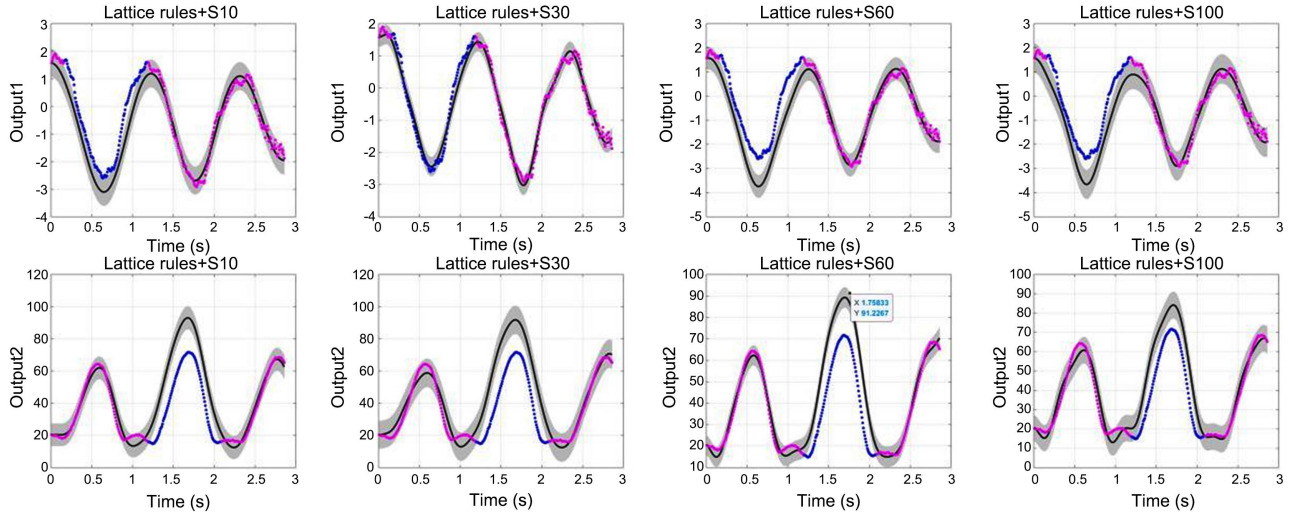

Figure 14. Predictive GPs with Lattice rules for walk data set.

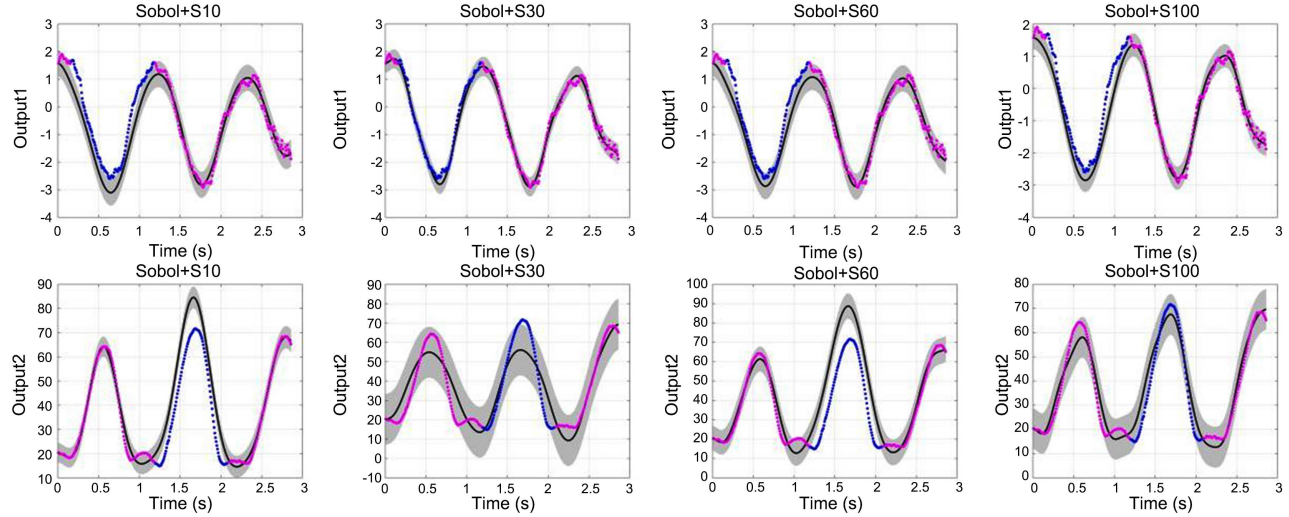

Figure 15. Predictive GPs with Sobol sequence for walk data set.

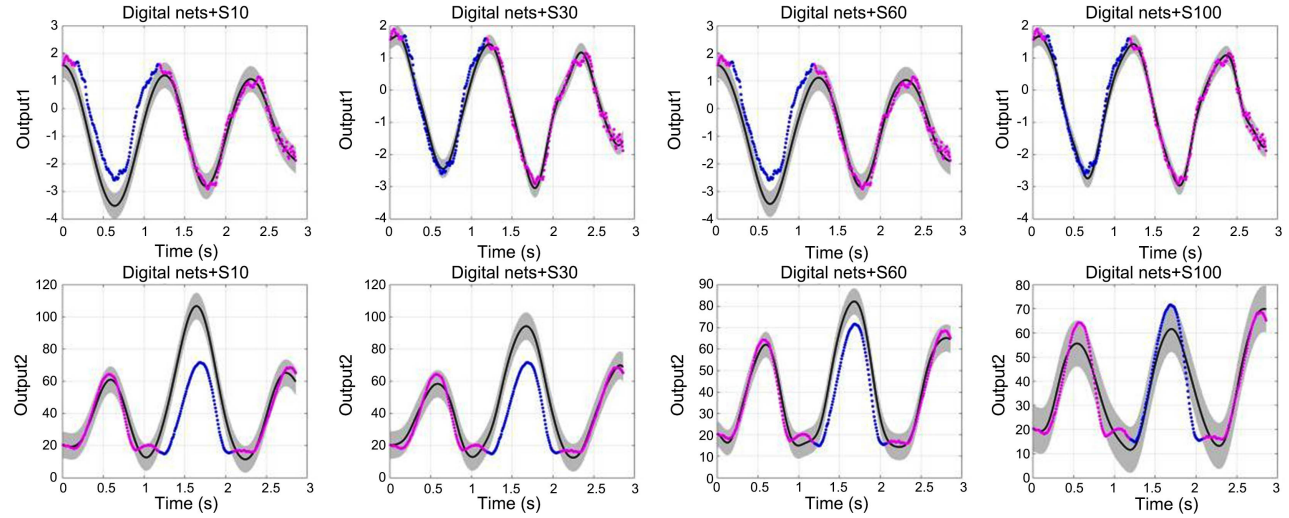

Figure 16. Predictive GPs with Digital nets for walk data set.

On the other hand, in the Walk data set for the second-order model, the data set is smooth and easy to learn, while the second-order LFM is relatively complex. Therefore, the data set usually only need 10 or 30 sampling points to fit, and more sampling points will lead to the overfitting problem. In this project, I mainly focus on the method to obtain good random Fourier features. The issue of overfitting or underfitting will not be in consideration. As a machine learning project, the purpose is to find a better solution for further use. Hence, each sequence's best performance will be recorded for comparison. The overfitting problem caused by the rise in the number of sampling points will not be considered this time.

For the experiments about the Swing data set, the smallest value for the MC method in lowerback-Yrot output is 0.93, which ranks 4^{th} among the five sequences. In the root-Ypos output, the MC method also ranks 4^{th} among them. This time, the Lattice rules in root-Ypos and the Sobol sequence in lowerback-Yrot are not better than the MC method. The Halton sequence and Digital nets in this experiment need fewer points but more accurate predictions. Especially the Halton sequence in lowerback output, the NMSE of which is 0.12, is much smaller than the MC method, whose NMSE is 0.93.

In the experiments about the Walk dataset, the model here is a second-order model, and the data set is smooth and regular. Hence, most of the sequences can have a good performance with few sampling points. Thus, for example, the result of the Halton sequence in lowerback-Yrot is 0.05, which is a litter higher than the MC method, whose NMSE is 0.04. But the result in Iradius-Xrot is different, and the Halton sequence got 0.01 while the MC method got 0.11. And the MC method in this experiment ranks both 4^{th} in lowerback-Yrot and Iradius-Xrot output. Thus, the Sobol sequence has the best performance in the lowerback-Yrot data set, and the Halton sequence has the best performance in Iradius-Xrot output.

In conclusion, because the sequences have different properties, and their performance may change according to the models and data sets. But the MC method always ranks 4^{th} or 5^{th} among all the five sequences, and there is no situation that the MC method ranks 1^{st}. Therefore, these experiments reveal that the QMC method is better than the original MC method in LFMs.

6.2. Goals Achieved

According to the analysis in Chapter3, this project has already implemented four kinds of low discrepancy sequences and obtained the random Fourier features. Moreover, via carrying out sixty experiments, the result is convincible and reasonable to show that the QMC method can further minimise the NMSE and NLPD.

7. Conclusions

In this dissertation, this project aims to obtain random Fourier features via a

more efficient sampling scheme. Therefore, this project consider taking five sequences to carry out the experiments, where four of the sequences belong to QMC method and the other belong to MC method.

In the experiments, they are evaluated according to NMSE and NLPD. Three datasets are used to sufficiently assess the performance for different kinds of data. Some datasets contain changeable data, while some datasets contain smooth data. For each dataset, each sequence is tested four times according to different numbers of sampling points. By a total of 60 experiments, the results are more reasonable and general to show the QMC method is a more efficient sampling scheme than the MC method.

This dissertation still has some shortcomings. More high-dimensional datasets can be applied in the experiment to test their performance in high-dimensional space. Besides, the project could also add more flexible and adaptive sequences to improve the performance even further.

Acknowledgements

My deepest gratitude goes first and foremost to Professor Dr Mauricio A Alvarez Lopez, my supervisor, for his constant encouragement and guidance. He has walked me through all the stages of the writing of this dissertation. Without his consistent and illuminating instruction, this dissertation could not have reached its present form.

Second, I would like to express my gratitude to the teachers who taught me in Sheffield, and they gave me much help and instructions over the past years.

Last, my thanks would go to my family for their considerations and support throughout these years.

Conflicts of Interest

The author declares no conflicts of interest regarding the publication of this paper.

References

[1] Alvarez, M., Luengo, D. and Lawrence, N.D. (2009) Latent Force Models. *Journal of Machine Learning Research*, **5**, 9-16.

[2] Gao, P., Honkela, A., Rattray, M. and Lawrence, N.D. (2008) Gaussian Process Modelling of Latent Chemical Species: Applications to Inferring Transcription Factor Activities. *Bioinformatics*, **24**, i70-i75. https://doi.org/10.1093/bioinformatics/btn278

[3] Alvarez, M.A., Luengo, D. and Lawrence, N.D. (2013) Linear Latent Force Models Using Gaussian Processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **35**, 2693-2705. https://doi.org/10.1109/TPAMI.2013.86

[4] Alvarez, M.A. and Lawrence, N.D. (2011) Computationally Efficient Convolved Multiple Output Gaussian Processes. *Journal of Machine Learning Research*, **12**, 1459-1500.

[5] Ghosh, S., Reece, S., Rogers, A., Roberts, S., Malibari, A. and Jennings, N.R. (2015)

Modeling the Thermal Dynamics of Buildings: A Latent-Force-Model-Based Approach. *ACM Transactions on Intelligent Systems and Technology* (*TIST*), **6**, 1-27. https://doi.org/10.1145/2629674

[6] Rahimi, A., Recht, B., *et al.* (2007) Random Features for Large-Scale Kernel Machines. *Conference on Neural Information Processing Systems*, **3**, 5.

[7] Williams, C.K. and Rasmussen, C.E. (2006) Gaussian Processes for Machine Learning. Volume 2, MIT Press, Cambridge. https://doi.org/10.7551/mitpress/3206.001.0001

[8] Wilson, A. and Adams, R. (2013) Gaussian Process Kernels for Pattern Discovery and Extrapolation. *International Conference on Machine Learning*, **28**, 1067-1075.

[9] Svensen, M. and Bishop, C.M. (2007) Pattern Recognition and Machine Learning. Springer, Berlin.

[10] Mitchell, T.M., *et al.* (1997) Machine Learning. McGraw-Hill Education, New York.

[11] Scholkopf, B., Smola, A. and Muller, K.-R. (1998) Nonlinear Component Analysis as a Kernel Eigenvalue Problem. *Neural Computation*, **10**, 1299-1319. https://doi.org/10.1162/089976698300017467

[12] Scholkopf, B., Mika, S., Burges, C.J.C., Knirsch, P., Muller, K.-R., Ratsch, G. and Smola, A.J. (1999) Input Space versus Feature Space in Kernel-Based Methods. *IEEE Transactions on Neural Networks*, **10**, 1000-1017. https://doi.org/10.1109/72.788641

[13] Muller, K.-R., Mika, S., Ratsch, G., Tsuda, K. and Scholkopf, B. (2001) An Introduction to Kernel-Based Learning Algorithms. *IEEE Transactions on Neural Networks*, **12**, 181-201. https://doi.org/10.1109/72.914517

[14] Vapnik, V. (2013) The Nature of Statistical Learning Theory. Springer Science & Business Media, Berlin.

[15] Vapnik, V.N. (1999) An Overview of Statistical Learning Theory. *IEEE Transactions on Neural Networks*, **10**, 988-999. https://doi.org/10.1109/72.788640

[16] Smola, A.J. and Scholkopf, B. (1998) Learning with Kernels, Volume 4. MIT Press, Cambridge.

[17] Shawe-Taylor, J., Cristianini, N., *et al.* (2004) Kernel Methods for Pattern Analysis. Cambridge University Press, Cambridge. https://doi.org/10.1017/CBO9780511809682

[18] Aizerman, M.A., Braverman, E.M. and Rozonoer, L.I. (1964) Theoretical Foundations of Potential Function Method in Pattern Recognition. *Automation and Remote Control*, **25**, 917-936.

[19] Mercer, J. (1909) Xvi. Functions of Positive and Negative Type, and Their Connection the Theory of Integral Equations. *Philosophical Transactions of the Royal Society of London. Series A*, **209**, 415-446. https://doi.org/10.1098/rsta.1909.0016

[20] Aronszajn, N. (1950) Theory of Reproducing Kernels. *Transactions of the American Mathematical Society*, **68**, 337-404. https://doi.org/10.1090/S0002-9947-1950-0051437-7

[21] Poggio, T. and Shelton, C.R. (2002) On the Mathematical Foundations of Learning. *Bulletin of the American Mathematical Society*, **39**, 1-49. https://doi.org/10.1090/S0273-0979-01-00923-5

[22] Scholkopf, B., Smola, A.J., Bach, F., *et al.* (2002) Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond. MIT Press, Cambridge.

[23] Wahba, G. (1990) Spline Models for Observational Data. SIAM, Philadelphia. https://doi.org/10.1137/1.9781611970128

[24] Cristianini, N., Shawe-Taylor, J., *et al.* (2000) An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods. Cambridge University Press, Cambridge. https://doi.org/10.1017/CBO9780511801389

[25] Wang, H.Q., Sun, F.C., Cai, Y.N., Chen, N., Ding, L.G., *et al.* (2010) Multi-Core Learning Method.

[26] Avron, H., Sindhwani, V., Yang, J.Y. and Mahoney, M.W. (2016) Quasi-Monte Carlo Feature Maps for Shift-Invariant Kernels. *The Journal of Machine Learning Research*, **17**, 4096-4133.

[27] Bochner, S. (1933) Monotone funktionen, stieltjessche integrale und harmonische analyse. *Mathematische Annalen*, **108**, 378-410. https://doi.org/10.1007/BF01452844

[28] Wang, F.S. and Lu, M.Y. (2013) Efficient Visual Tracking via Hamiltonian Monte Carlo Markov Chain. *Computer Journal*, **56**, 1102-1112. https://doi.org/10.1093/comjnl/bxs141

[29] Mazhdrakov, M., Benov, D. and Valkanov, N. (2018) The Monte Carlo Method: Engineering Applications. ACMO Academic Press, Cambridge.

[30] Metropolis, N., *et al.* (1987) The Beginning of the Monte Carlo Method. *Los Alamos Science*, **15**, 125-130.

[31] Rubinstein, R.Y. and Kroese, D.P. (2016) Simulation and the Monte Carlo Method. Volume 10, John Wiley & Sons, Hoboken. https://doi.org/10.1002/9781118631980

[32] Feller, W. (1957) An Introduction to Probability Theory and Its Applications. Wiley, Hoboken.

[33] Press, W.H., Press, W.H., Flannery, B.P., Teukolsky, S.A., Vetterling, W.T., Flannery, B.P. and Vetterling, W.T. (1989) Numerical Recipes in Pascal: The Art of Scientific Computing. Volume 1, Cambridge University Press, Cambridge.

[34] Asmussen, S. and Glynn, P.W. (2007) Stochastic Simulation: Algorithms and Analysis. Volume 57, Springer Science & Business Media, Berlin. https://doi.org/10.1007/978-0-387-69033-9

[35] Hickernell, F.J. (2014) Koksma-Hlawka Inequality. Wiley StatsRef: Statistics Reference Online. https://doi.org/10.1002/9781118445112.stat03070

[36] Niederreiter, H. (1992) Random Number Generation and Quasi-Monte Carlo Methods. SIAM, Philadelphia. https://doi.org/10.1137/1.9781611970081

[37] van der Corput, J.G. (1935) Verteilungsfunktionen. I. *Proceedings of the Koninklijke Nederlandse Akademie van Wetenschappen,* Amsterdam, Vol. 38, 813-821.

[38] Gelman, A., Hwang, J. and Vehtari, A. (2014) Understanding Predictive Information Criteria for Bayesian Models. *Statistics and Computing*, **24**, 997-1016. https://doi.org/10.1007/s11222-013-9416-2

[39] http://www.bramblemet.co.uk

[40] Nguyen, T.V., Bonilla, E.V., *et al.* (2014) Collaborative Multi-Output Gaussian Processes. *Proceedings of the Thirtieth Conference on Uncertainty in Artificial Intelligence*, *UAI* 2014, Quebec City, 23-27 July 2014, 643-652.

[41] (2021) http://mocap.cs.cmu.edu/

[42] Kocis, L. and Whiten, W.J. (1997) Computational Investigations of Low-Discrepancy Sequences. *ACM Transactions on Mathematical Software* (*TOMS*), **23**, 266-294. https://doi.org/10.1145/264029.264064

[43] Kuo, F.Y. and Nuyens, D. (2016) Application of Quasi-Monte Carlo Methods to Elliptic Pdes with Random Diffusion Coefficients: A Survey of Analysis and Imple-

mentation. *Foundations of Computational Mathematics*, **16**, 1631-1696. https://doi.org/10.1007/s10208-016-9329-5

[44] Joe, S. and Kuo, F.Y. (2008) Constructing Sobol Sequences with Better Two-Dimensional Projections. *SIAM Journal on Scientific Computing*, **30**, 2635-2654. https://doi.org/10.1137/070709359

[45] Dick, J., Kuo, F.Y. and Sloan, I.H. (2013) High-Dimensional Integration: The Quasi-Monte Carlo Way. *Acta Numerica*, **22**, 133-288. https://doi.org/10.1017/S0962492913000044