Chapter 1

Emulation of LAMMPS outputs

We describe the procedure for building a simple emulator of LAMMPS output.

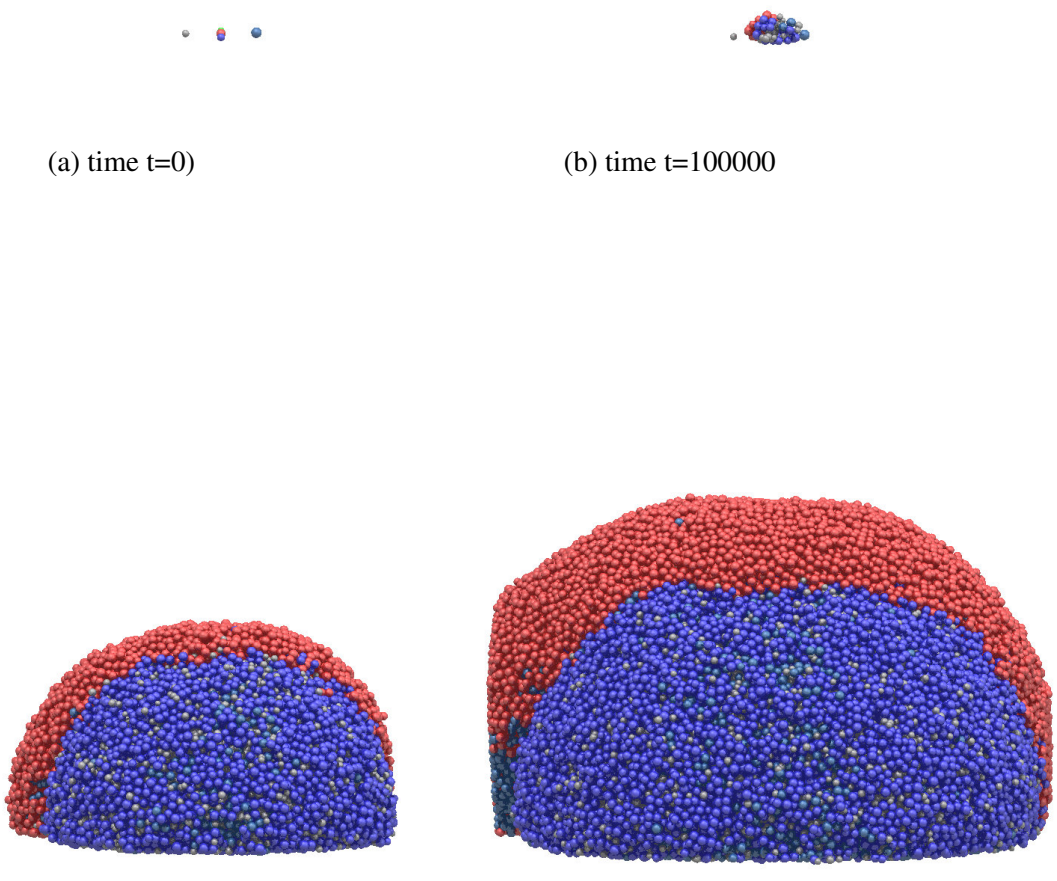
1.1 Experimental design

This section describes the procedure for generating the parameter combinations and variables at which the LAMMPS model is run. We run the LAMMPS code for a small sample of inputs using a Latin Hypercube Design (LHD). This produces data for training the LAMMPS emulator to approximate the major LAMMPS outputs. LHD provides a good coverage of the input space with a relatively small number of design points. We use maximin LHS technique that optimises samples by maximizing the minimum distance between design points Santner et al. (2003). Suppose we want to sample a function of p variables, the range of each variable is divided into n probable intervals, n sample points are then drawn such that a Latin Hypercube is created.

We generate an $n \times p$ variables Latin Hypercube sample matrix with values uniformly distributed on interval [0,1]. We then transformed the generated sample to the quantile of a uniform distribution using the range of the parameters given in Table 4.1. LAMMPS model is computationally demanding, we limit our initial analysis to just $n = 100$ training points for each dimension of the $p = 22$ input spaces that are to be varied.

1.1.1 Simulation data

We describe two different simulation procedures. Firstly, let the design matrix which contain the input to the LAMMPS model be denoted by $D = (\theta_p^i, t_p, p = 1,\ldots,22; i = 1,\ldots,100)$, where the subscript p represents the 22 input parameters and superscript i denote 100 different realisations (design points), t_p is the time in seconds at which the output data is recorded. The design matrix $D_{100\times 23}$ denotes the input values at which the LAMMPS model will be run for every combinations of x_p where x_p represents



(a) time t=0)

(b) time t=100000

(c) time t=300000

(d) time t=350,000

Figure 1.1: Sample of LAMMPS simulation data for particle growth for 4 different time points (seconds).

p^{th} row of D. The current LAMMPS code is set up to produce the following outputs namely particle diameter, position (3-dimensional), velocity (3-dimensional) and force

(3-dimensional). The LAMMPS code could be run for as long as there are sufficient computing resources to store the outputs. For the present analysis, the code is run for 352800 seconds to generate sufficient data for the emulation which is equivalent to ≈ 4 days of real time and generated output results are saved at a time-step of 2000 seconds which gives about 176 different time slices.

Therefore, a single run of the code for each of design point x_p will produce ten different outputs and 176-time steps. There are five different types (or species) of the particle in the simulation namely AOB, NOB, HET, EPS and inert. We note that the shape and number of particles as well as the composition of the floc at each time step vary in the simulation.

We performed the second simulation using the same input configuration as described above but repeated the runs for five times to incorporate stochastic variations in our outputs. This, of course, increases the amount of CPU time for the entire simulations even with running the bash script in parallel on a Linux machine.

1.2 Method 1: Combination of linear models and Gaussian process regression

1.2.1 Procedure

We describe the analysis of the first dataset here. The work focuses on predictions of mean floc diameter from LAMMPS simulation outputs together with their associated uncertainty levels. We propose a two-stage approach similar to O'Hagan (2006) where we shall combine a linear model in the first stage and a Gaussian process regression for residual interpolation in the second stage.

The step by step procedure for emulating one of the outputs of LAMMPS model, "particle floc diameter" is given here. We start by computing the average diameter of all the particles at each time step to obtain a vector $\mathbf{y}(x) = \bar{z}_1, \ldots, \bar{z}_{176}$ of the mean floc diameter to be emulated such that

$$\bar{z}_t = \frac{\sum_{k=1}^{N} z_{kt}}{N} \tag{1.1}$$

where $k = 1, \ldots, N$, N represents the total number of all 5 particles at each time slice. We also note that N varies across time point and in particular increases with time as seen in Figure 1.1 and z_{kt} is a simulation floc diameter for particle k at time t.

1.2.2 Stage 1: Linear models

For the training data, we sub-sample 75 out of 176-time slices for each design point x_p which gives a total of 7500 observations. The training data are (7500×1) vector of mean floc diameter $\mathbf{y}$ of LAMMPS output and (7500×23) matrix $\mathbf{X}$, which holds the values taken by the explanatory variables. (NOTE: sequence of time t at which output is sought is used as an additional input to the emulator). We fit a linear model to data $[\mathbf{y}, \mathbf{X}]$ and then obtain the predictions $\mu^{\bullet}$. We also use the fitted model to predict the left out observations $\tilde{\mathbf{y}}^{new}$ for cross-validation purpose and to obtain standard error of predictions $\mathbf{s}^{new}$, which is a measure of prediction uncertainty in the linear model.

In order to construct emulator for the prediction of mean particle diameter, we apply a standard regression to the data in the first stage using equation (1.2) and use a Gaussian process regression for interpolating unexplained residual from linear model

$$\mathbf{y} = f(\mathbf{x}) + \varepsilon = \beta_0' + \beta_1' x_1 + \ldots + \beta_p' x_p + \varepsilon \tag{1.2}$$

where $\mathbf{y}$ is the LAMMPS simulated mean floc diameter, $p = 23$ is the number of parameters for estimation and $\mathbf{x}_1, \ldots, \mathbf{x}_p$ are independent variables and $\beta_1', \ldots, \beta_{23}'$ are regression coefficients. We assume $\varepsilon \sim N(0, \sigma^2)$.

1.2.3 Summary of steps taken for stage 1 modelling

(i) Design of experiment to identify important outputs (which outputs are to be emulated).

(ii) Determine the relevant input factors to be varied (use all the 22 factors for now).

(iii) Assign uniform probability density to each input variable (we have little knowledge about these parameters).

(iv) Generate LHS of 100 design points on each of the 22 input parameters and transform into the quantile of uniform distribution using parameter ranges in Table 4.1.

(v) Evaluate LAMMPS model for various input combinations to obtain the training data.

(vi) Compute the LAMMPS mean floc diameter for each time-step from output data

(vii) Randomly sample 75 time points out of 176 for each design point

(viii) Fit a linear model with equation (1.2) using all 23 parameters (time is included as an additional variable to the linear model)

(ix) Obtain both predictions and standard error of predictions for all data points

1.2.4 Stage 2: GP modelling of residual data

The (7500×1) vector of residual $\varepsilon = \mathbf{y} - \mu^{\bullet}$ obtained from stage 1 formed our new training data in this section. This residual can be modelled as $\varepsilon = \eta(X')$. We do not apply GP to all the 23 input variables in stage 1. Rather, we use standardized regression coefficients of equation (1.2) as a measure of sensitivity to select important explanatory variables for the GP regression such that $X' << X$ (ie p is now < 23). Since Gaussian process is described completely by its mean and covariance functions, the mean function is given as

$$E[\eta(X')|\beta] = h^{\mathrm{T}}(\mathbf{X}')\beta, \tag{1.3}$$

where $h(\mathbf{X}')$ is a vector of regression functions. For our analysis, we use a simple linear function $h(\mathbf{X}') = (1, \mathbf{x}^T)$ and β is an unknown hyperparameter to be estimated and covariance function

$$K = Cov[\eta(\mathbf{x}), \eta(\mathbf{x}')|\sigma^2, \alpha] = \sigma^2 \mathbf{C}(\mathbf{x}, \mathbf{x}^{\mathrm{T}}),$$

where σ^2 is a noise variance and $\mathbf{C}(\mathbf{x}, \mathbf{x}^{\mathrm{T}})$ is a correlation function with an hyperparameter α. Gaussian and exponential correlation functions of the form

$$\mathbf{C} = \Big\{ \exp(-(x - x')^T \alpha (x - x')) \Big\}, \tag{1.4}$$

$$\mathbf{C} = \exp(-(x - x')/\alpha) \tag{1.5}$$

where α is the correlation hyperparameters to be estimated from the data. It is difficult to apply a GP directly to the entire residual data because of the inversion of covariance matrix which scales cubically with the number of observations $\mathrm{O}(N^3)$, we proceed by sub-sampling just 200 random observations from the data and denote the training data as $[\mathbf{y}, \mathbf{X}']$, where $\mathbf{y} = [\varepsilon_1 = \eta(x_1), \ldots, \varepsilon_{200} = \eta(x_{200})]$. We select just 4 important explanatory variables for the GP modelling and our correlation parameter is denoted as $\alpha = (\alpha_1, \ldots, \alpha_4)$.

For computing the posterior distribution, we assign a non-informative prior to the β, σ^2 and α parameters such that $p(\beta, \sigma^2) \sim \sigma^{-2}$ and $p(\alpha) \sim 1$. We update these prior distributions on some data $\mathbf{y}$ to get the posterior estimates using Bayes theorem. We estimate their values by conditioning on each parameter and maximize the resulting marginal likelihood function (Andrianakis & Challenor, 2009; Hankin, 2005). Therefore, the resulting posterior distribution is denoted as $\eta(.)|\mathbf{y} \sim N\Big\{\mu^{\star}, K^{\star}\Big\}$, where

$$\mu^{\star}(\mathbf{x}) = h(\mathbf{x})^T\hat{\beta} + t(\mathbf{x})^T\mathbf{C}^{-1}[\mathbf{y} - \mathbf{H}\hat{\beta}] \tag{1.6}$$

$$\begin{aligned} K^{\star} = \hat{\sigma}^2\Big\{c(\mathbf{x},\mathbf{x}') - t(\mathbf{x})^T\mathbf{C}^{-1}t(\mathbf{x}) + \\ \Big(h(\mathbf{x})^T - t(\mathbf{x})^T\mathbf{C}^{-1}t(\mathbf{x})\Big)(\mathbf{H}^T\mathbf{C}^{-1}\mathbf{H})^{-1}\Big(h(\mathbf{x}')^T - t(\mathbf{x}')^T\mathbf{C}^{-1}t(\mathbf{x}')\Big)^T\Big\} \end{aligned} \tag{1.7}$$

where $t(\mathbf{x})^T = [c(\mathbf{x},\mathbf{x}_1),\ldots,c(\mathbf{x},\mathbf{x}_n)]$ and $\mathbf{H}^T = [h(\mathbf{x}_1),\ldots,h(\mathbf{x}_n)]$, see more details in the appendix.

Finally, let the GP posterior mean and variance estimates of the residual at a new input point x^{new} (cross-validation) be denoted as $\mu^{\star\star}$ and $K^{\star\star}$ respectively, and predictions and standard error of predictions from linear model as $\tilde{\mathbf{y}}^{new}$ and $\mathbf{s}^{new}$ respectively. The final predictions of mean particle diameter $\mu^{\bullet}(x)^{new}$ is given as

$$\mu^{\bullet}(x)^{new} = \tilde{\mathbf{y}}^{new} + \mu^{\star\star}(\mathbf{x}) \tag{1.8}$$

and total uncertainty is

$$\hat{\mathbf{s}}^{new} = \mathbf{s} + \sqrt{|K^{\star\star}|} \tag{1.9}$$

1.2.5 Summary of stage 2

(i) Perform the GP emulation of the unexplained residual data

(ii) Given the posterior density of α as defined in equation (A.10c) of Appendix 1, compute the MLE of α, such that equation (A.10c) is maximized.

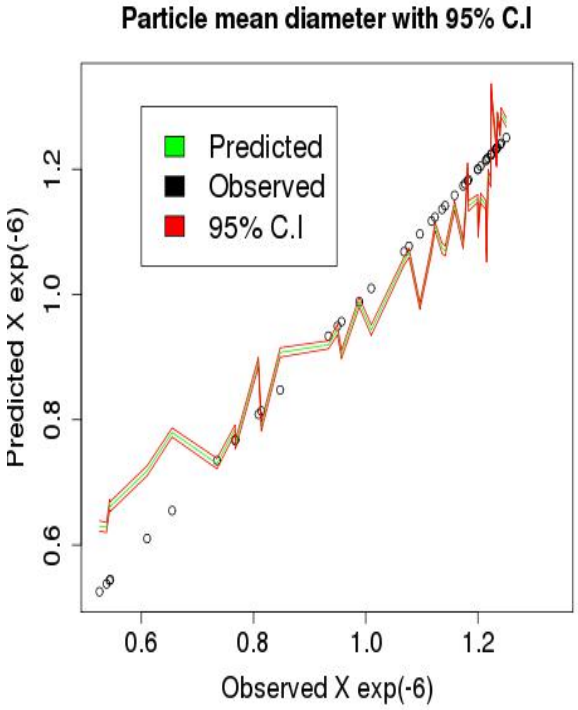
(iii) Set $\mathbf{C} = \mathbf{C}(\hat{\alpha})$

(iv) Compute estimate $\hat{\beta} = (\mathbf{H}^T\mathbf{C}^{-1}\mathbf{H})^{-1}\mathbf{H}^T\mathbf{C}^{-1}\mathbf{y}$

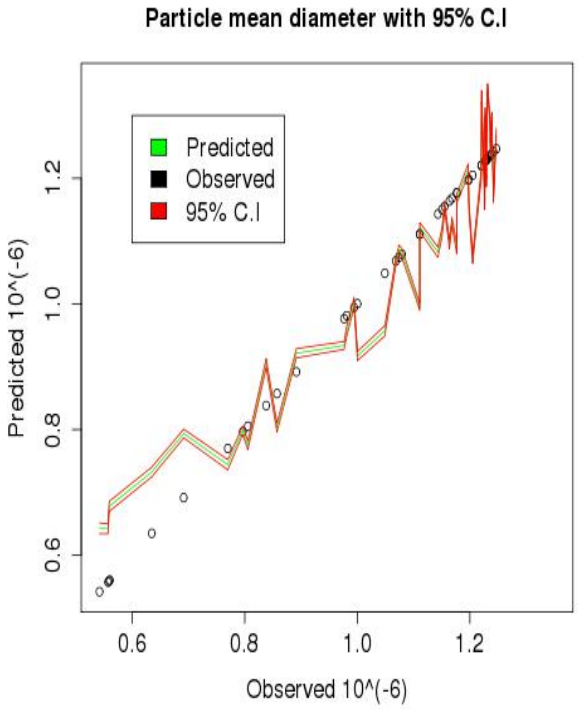
(v) Compute estimate $\widehat{\sigma^2} = \frac{1}{n-p}\Big[(\mathbf{y} - \mathbf{H}\hat{\beta})^T\mathbf{C}^{-1}(\mathbf{y} - \mathbf{H}\hat{\beta})\Big]$

(vi) Compute posterior mean estimate $\mu^{\star}(\mathbf{x})$ given in equation (1.6)

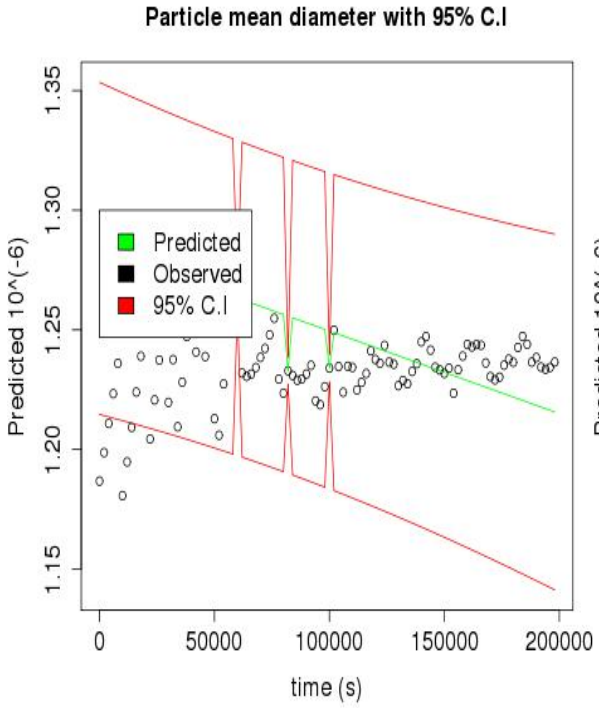
(vii) Compute posterior variance estimate $K^{\star}$ given in equation (1.7) above



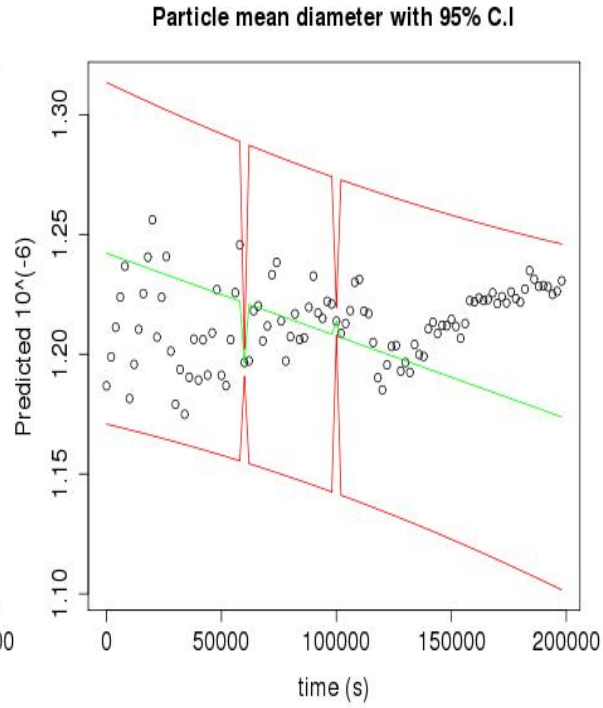
(a) LAMMPS and model predictions (emulator) for some randomly selected points



(b) LAMMPS and model predictions (emulator) for some randomly selected points



(c) LAMMPS and model predictions at 79^{th} design point for all time steps



(d) LAMMPS and model predictions at 28^{th} design point for all time steps

Figure 1.2: Pair plot of model predictions (emulator) vs. LAMMPS values with their 95% C.I for mean floc diameter (all species) from a two-stage technique

Chapter 2

Method 2: Kriging

Having obtained the simulation data by running the LAMMPS model as described in the previous section, we describe an alternative emulation approach for fitting LAMMPS output data. This approach involves using a kriging model, where the two stage techniques describe in previous section are combined as a single step. Kriging is a geostatistical technique for interpolating the value of an unknown random observation from data $\mathbf{y}(\mathbf{x})$ observed at known locations. Kriging models are also commonly used for building cheaper surrogate model of expensive computer codes Currin et al. (1991); Martin & Simpson (2004); Osio et al. (1996); Li & Sudjianto (2005). Here, $\mathbf{y}(\mathbf{x})$ can be decomposed into a mixture of deterministic (non-random trend) and a residual random variation. The trend could be modelled as a constant in ordinary/simple kriging or as an n^{th} order polynomial in universal kriging. In this section, we shall discuss the universal kriging technique. The model formulation is given as

$$\mathbf{y}(\mathbf{x}) = f(\mathbf{x}) + \varepsilon(\mathbf{x}) \tag{2.1}$$

where $\mathbf{y}(\mathbf{x})$ is as defined in equation (1.2) above. The deterministic function $f(\mathbf{x})$ is the mean approximation of the expensive computer simulator (eg LAMMPS) and f is a polynomial function. Under this assumption, $f(x)$ can be modelled as

$$f(\mathbf{x}) = \sum_{j=1}^{p} \beta_j h_j(x) = \mathbf{H}(x)\beta \tag{2.2}$$

$\beta = [\beta_1, \ldots, \beta_p]$ is a $(p \times 1)$ vector of unknown regression coefficients and $\mathbf{H}(x) = [h_1(x), ..., h_p(x)]^T$ is a $(n \times p)$ matrix of regression functions, $\varepsilon(\mathbf{x})$ is a stochastic Gaussian process with mean zero and characterize by its covariance function $K = Cov(\varepsilon(\mathbf{x}), \varepsilon(\mathbf{x})) = \sigma^2 \mathbf{C}(\mathbf{x}, \mathbf{x}')$, where σ^2 denotes the variance of $\varepsilon(\mathbf{x})$ also called process variance and $\mathbf{C}$ is a $(n \times n)$ positive definite matrix of correlation between $\varepsilon(\mathbf{x})$'s at the experimental design points. We are assuming a univariate output and a deterministic computer model. Similarly, $\mathbf{t}(x^{new}) = [Cor(x_1, x^{new}), \ldots, Cor(x_n, x^{new})]^T$ for

the $(n \times 1)$ vector of correlations between the $\varepsilon(\mathbf{x})$'s at the design points and new input points x^{new}. We use both Gaussian equation and exponential correlation functions, equations (1.4,1.5) Sacks et al. (1989); Kleijnen (2009); Kleijnen & Simpson (2005).

The universal kriging predictor $\mu^{\bullet}(x^{new})$ of the value of $\mathbf{y}(x)$ at the new target point x^{new} is the linear predictor

$$\mu^{\bullet}(x^{new}) = \sum_{j=1}^{n} \lambda_j \mathbf{y}(x_j) = \lambda(x)^T \mathbf{y}, \tag{2.3}$$

for the sample points $x_1, \ldots, x_n$, where the coefficients or weights $\lambda = (\lambda_1, \ldots, \lambda_n)^T$ are estimated by minimizing the variance of prediction error for each realizations of the random function $\mathbf{y}(x)$. The best linear unbiased predictor (BLUP) is computed by minimizing the mean squared error

$$MSE[\mu^{\bullet}(x^{new})] = E[\lambda^T \mathbf{y} - \mathbf{y}(x^{new})]^2, \tag{2.4}$$

subject to the unbiasedness constraint $E[\lambda^T(x)\mathbf{y}] = E[\mathbf{y}(x^{new})]$. The mean squared error in equation (2.4) can be rewritten by substituting the value of $\mathbf{y}$ in equation (2.1), we thus have

$$\sigma^2[1 + c\lambda^T(x)\mathbf{C}\lambda(x) - 2\lambda^T(x)\mathbf{t}(x^{new})], \tag{2.5}$$

The unbiasedness constraint is now denoted as $H^T\lambda(x) = h(x)$. The optimal weights λ_j^* in $\mathbf{y}(x^{new})$ is estimated by using Lagrange multipliers to constraining MSE minimization. The Lagrange multiplier is also use to solve system of equations in order to compute p coefficients of λ. Given the inverse of matrix $\mathbf{C}$, then the best linear unbiased predictor for kriging model is given as

$$\mu_{uk}^{\bullet}(x) = h^T(x)\hat{\beta} + \mathbf{t}^T(x)\mathbf{C}^{-1}(\mathbf{y} - \mathbf{H}\hat{\beta}) \tag{2.6}$$

Similarly, the variance follows by substituting in the optimal value of $\lambda^*(x)$ in the MSE equation, we have

$$\mathbf{K}_{uk}^{\bullet} = \hat{\sigma}^2 \Big\{ C(\mathbf{x}, \mathbf{x}') - \mathbf{t}(\mathbf{x})^T \mathbf{C}^{-1} \mathbf{t}(\mathbf{x}) + \Big(h(\mathbf{x})^T - \mathbf{t}(\mathbf{x})^T \mathbf{C}^{-1} \mathbf{t}(\mathbf{x}) \Big) (\mathbf{H}^T \mathbf{C}^{-1} \mathbf{H})^{-1} \Big(h(\mathbf{x}')^T - \mathbf{t}(\mathbf{x}')^T \mathbf{C}^{-1} \mathbf{t}(\mathbf{x}') \Big)^T \Big\}. \tag{2.7}$$

See more details in (Sacks et al., 1989; Santner et al., 2003; Kleijnen & Mehdad, 2014)

The next problem is how to estimate the unknown parameters. We use a Maximum Likelihood Estimation (MLE) technique like many other authors (Santner et al., 2003; Kleijnen & Mehdad, 2012), are being used as an estimator of the kriging model parameters because of its computational efficiency, $\theta = (\beta, \sigma^2, \alpha)$. MLE is based on the assumption of Gaussian probability distribution. The likelihood of the model param-

eters is defined as the probability of the n observations $\mathbf{y} = y_1, , y_n$, given the model parameters such that

$$L(\theta|\mathbf{y}) = \Pi_{j=1}^{n} p(\mathbf{y}_j|\theta). \tag{2.8}$$

The expression for $L(\theta|\mathbf{y})$ is given by equation below as

$$L(\beta, \sigma^2, \alpha; \mathbf{y}) \propto \frac{|\mathbf{C}|)^{-\frac{\mathbf{1}}{\mathbf{2}}}}{(2\pi\sigma^2)^{\frac{n}{2}}} \exp\left\{ \frac{(\mathbf{y} - H\beta)^T \mathbf{C}^{-\mathbf{1}} (\mathbf{y} - \mathbf{H}\beta)}{2\sigma^2} \right\} \tag{2.9}$$

where $\alpha = [\alpha_1, \ldots, \alpha_n]$ is a vector of correlation lengths and $|\mathbf{C}|$ is the determinant of correlation matrix $\mathbf{C}$ and $K = \sigma^2\mathbf{C}$. By taking the derivative of the log-likelihood of equation (2.9) with respect to β and σ^2 and solving for zero, the estimates $\hat{\beta}$ and $\hat{\sigma}^2$ are given respectively as $\hat{\beta} = (\mathbf{H}^T\mathbf{C}^{-1}\mathbf{H})^{-1}\mathbf{H}^T\mathbf{C}^{-1}\mathbf{y}$ and $\hat{\sigma^2} = \frac{1}{n}\left[(\mathbf{y} - \mathbf{H}\hat{\beta})^T \mathbf{C}^{-\mathbf{1}} (\mathbf{y} - \mathbf{H}\hat{\beta})\right]$. The alternative way of performing this computation under fully-Bayesian technique is to marginalise the conditional $(p(.)|\mathbf{y}, \beta, \sigma^2, \alpha)$ with respect to posteriors of β and σ^2. In order to estimate α, we maximize over α the concentrated likelihood given below after plugging the values of $\hat{\beta}$ and $\hat{\sigma}^2$

$$-2logL(\hat{\beta}, \hat{\sigma}^2, \alpha; \mathbf{y}) = nlog(2\pi) + nlog\hat{\sigma}^2 + log(|\mathbf{C}|) + n \tag{2.10}$$

We observe that that both $\hat{\sigma}^2$ and $\mathbf{C}$ depend upon the correlation parameter α. The trend and covariance parameters θ are computed quickly and very efficiently by using a global optimiser which is based on the extension of the efficient algorithm proposed in Park & Baek (2001) for likelihood maximization. See further details in (Roustant et al., 2012).

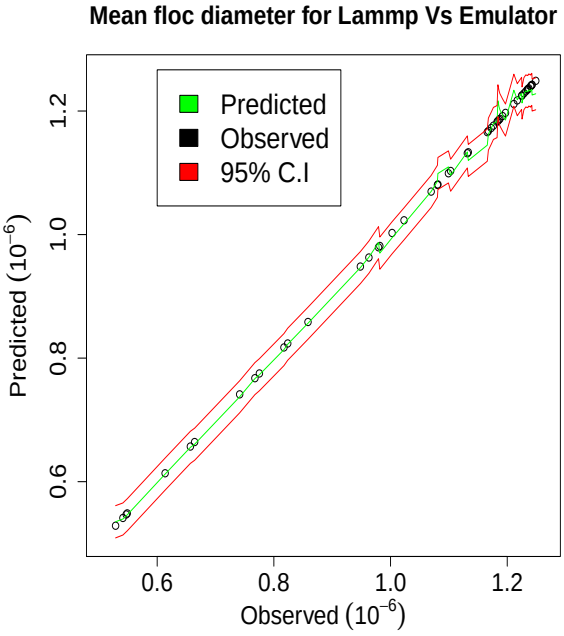
For the noisy observations, covariance $K = \sigma^2\mathbf{C}$ is replaced by $\sigma^2\mathbf{C} + \tau^2\mathrm{I}$ in equations (2.9,2.7,2.6) respectively, where $\tau^2 = \tau_1^2, \ldots, \tau_n^2$ are the noise variances and I is a diagonal matrix of ones. Kriging technique is equivalent to the Bayesian method describes in section (1.2.4) if we assign improper uniform priors on the β. In other words, non-informative Bayesian analysis often leads to kriging predictor and variance and these estimators appear respectively as conditional mean and variance in equations (1.6 and 1.7).

2.1 Kriging results

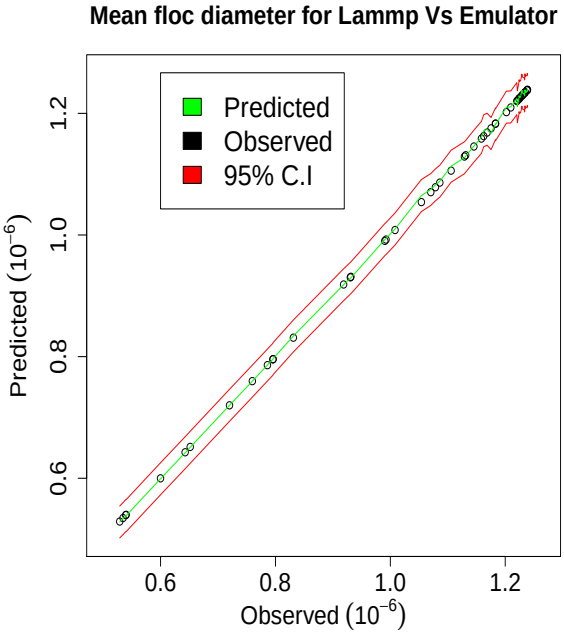
Some results from kriging technique are given here. Recall from above that our training data are (7500×1) vector of mean floc diameter $\mathbf{y}$ of LAMMPS output and (7500×23) matrix $\mathbf{X}$ of input variables. Here, we use the same 75 out of 176-time slices but we sub-sample just 50 design points to reduce further the dimension of the data which gives a total of 3750 observations. The new training data for the kriging model is

$D = [\mathbf{y}_{3750\times1}, X_{3750\times23}]$. We consider only the linear terms of the trend model (no quadratic terms). We compare both Gaussian and exponential covariance functions with little difference in their predictions. The performance of the kriging emulator is tested by using the fitted model to predict the left out data.

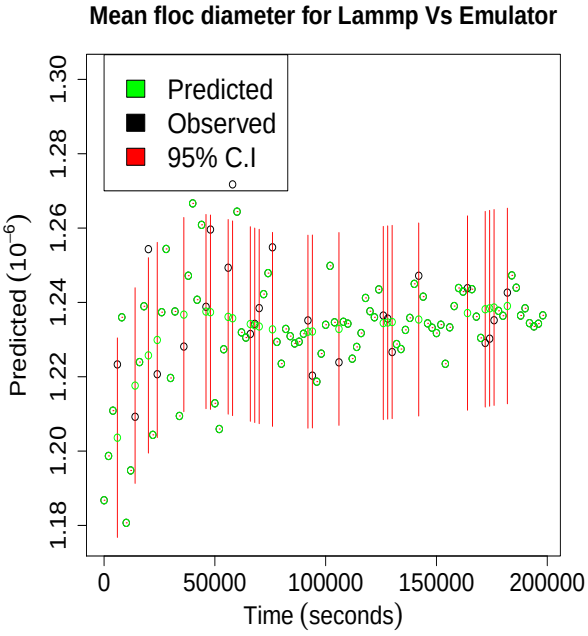
We also perform the sensitivity analysis of the mean particle floc diameter to identify relevant variables. We use a Sobol technique which involves decomposing total output variance as a summand of increasing dimensionality. We sampled 1000 observations from a uniform distribution for each of the 23 input variables in Table 1. We compute the sensitivity indices for three different methods. Bootstrapping was used to compute 95% confidence intervals on the estimated indices. The indices are shown in Figure 2.2 and we can see that only four parameters are relevant. Finally, we refitted the kriging model using just these four important parameters, some visual plots of results comparing the LAMMPS simulations with the emulator are showing in Figure 2.1. We also compute the total percentage of variance explained by the final model which is $\sim 99\%$.



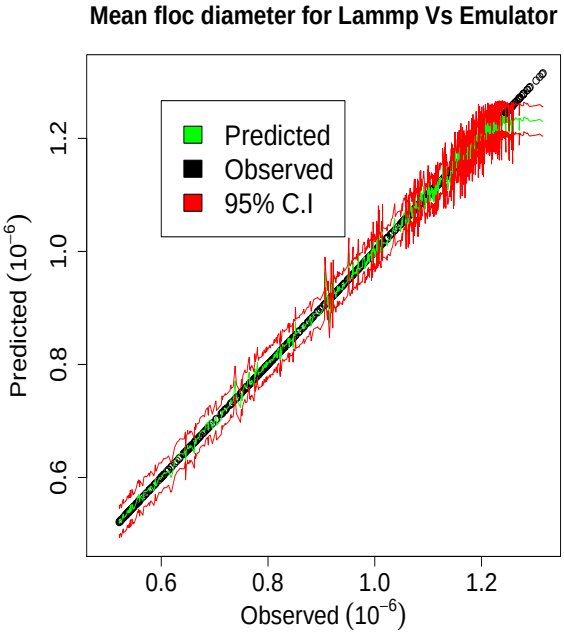
(a) LAMMPS and kriging predictions (emulator) for some randomly selected points



(b) LAMMPS and kriging predictions (emulator) for some randomly selected points



(c) LAMMPS and kriging predictions at 20^{th} design point for all time steps. Note: The points with no C.I bands are the design points where MSE=0



(d) Pair plot for complete data set

Figure 2.1: Pair plot of kriging predictions (emulator) vs. LAMMPS values with their 95% C.I for mean floc diameter (all species) using a Gaussian covariance functions

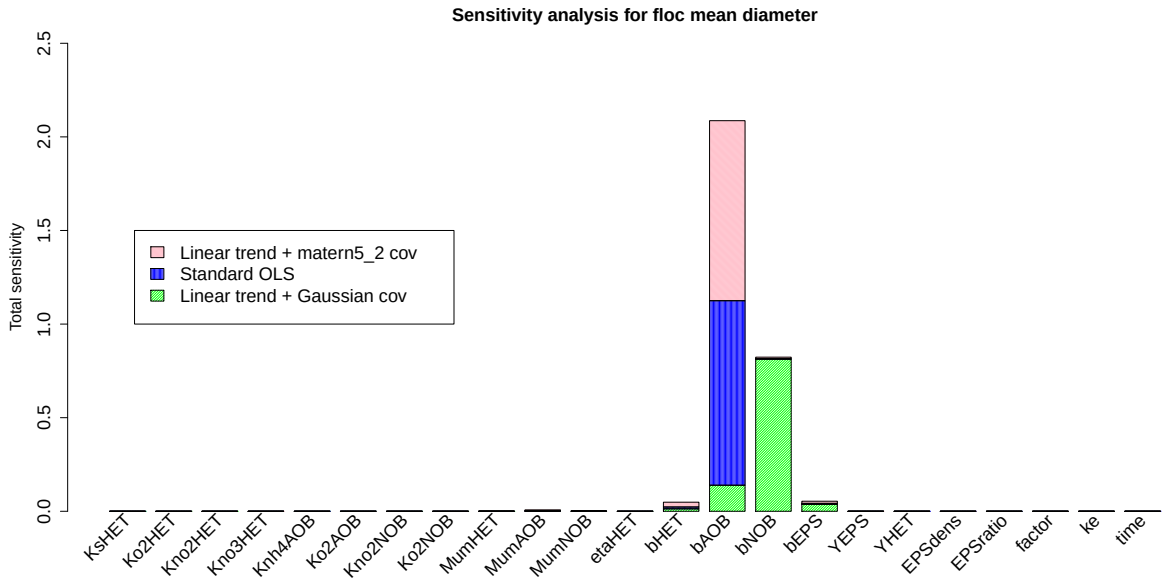


Figure 2.2: Barplots showing total sensitivity indices for mean floc diameter using Sobol based variance decomposition technique. Comparison for three different methods.

Chapter 3

Emulation of floc and biofilm

3.1 Introduction

Flocs are an aggregation of microbes mixed with an adhesive material called EPS. They are often difficult to measure or quantify because of their irregular size and shape. A wide range of different equivalent diameters is often used to characterize the floc size, see Jarvis et al. (2005) for further details. The individual particle that makes up the flocs is simulated as a sphere of a variable volume. Our approach is to focus on the cluster of particles as a floc because of a vast number of data involve and emulate their interested properties stated below. The floc is treated as a ball of a sphere, and we estimate the diameter of a sphere that circumscribes its boundary/outline. The center of the sphere will be equivalent to the center of mass of the component particles as shown in Figure (3.1). The detailed procedure of emulating the floc diameter and (biofilm) will be described in this section. In the further analysis, we will also emulate the center of the sphere to give spatial attributes to the floc characterization.

3.2 Simulation

Let the design matrix which contain the input to the LAMMPS model be denoted by $\mathbf{X} = (\theta_p^i, t, p = 1,\ldots,25; i = 1,\ldots,1000)$, where the subscript $p = 25$ represents 20 callibrated model parameters and 5 variables that represent the model initial conditions (see Table 4.1), superscript i denote the 1000 different realisations (design points) and t is the time slice in seconds at which the output data is recorded $t = 1,\ldots,T$. The design matrix $\mathbf{X}_{1000\times 26}$ denotes the input values at which the LAMMPS model is run for every combinations of x_i which is a point in $\mathbf{X}$, where x_i represents i^{th} row of $\mathbf{X}$. The LAMMPS code is run for two days (172800 s simulation time).

The simulations are repeated 100 times for each design point i to incorporate stochastic variations in our outputs. The simulations are recorded at a time-step of

2000 seconds which gives about 86 different time slices. This will provide initial estimates of the mean and variance of the LAMMPS model.

The current LAMMPS code is set up to produce the following outputs namely particle diameter, mass, position (3-dimensional) for each time step t. The time series output at each design point is denoted as a matrix $\mathbf{Y} = [\mathbf{y}_1, \ldots, \mathbf{y}_T]$, such that $\mathbf{y} = [y_1, \ldots, y_n]$, where $T = 86$ in this simulation and n is the total number of particles at each time step. The number of particles n at each time slice varies across the design points and, in particular, increasing with time as it expected for the microbes to be growing. We make another independent simulation of 100 runs with ten replicates for cross-validation purpose. Here, the simulation is run for a longer period than the previous simulations (4 days simulation). We consider emulation of floc which is summarized by aggregating all the individual microbe at each time step.

3.3 Procedure for emulating floc equivalent diameter

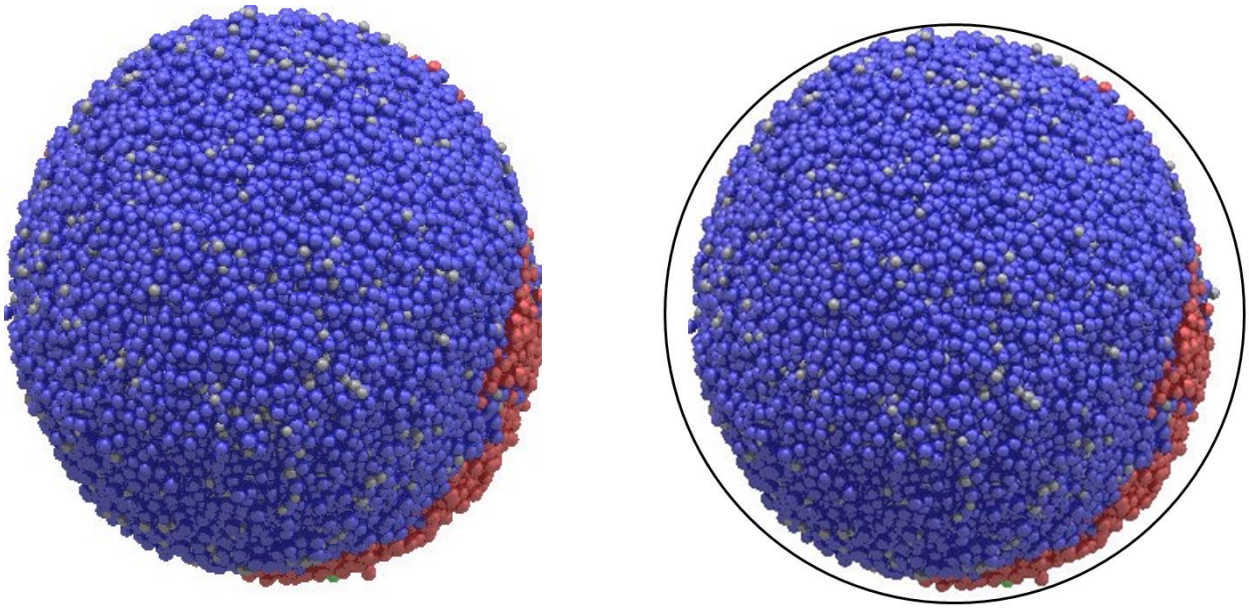


Figure 3.1: Transformation of microscale particles to floc at the mesoscale. Floc equivalent diameter is the diameter of the smallest sphere that circumscribes the outline of the projected floc.

3.4 The challenges

Some of the main challenges of LAMMPS emulation are the nature of the outputs produced from the model that make it much difficult to emulate

(1) The LAMMPS model is expensive to evaluate - we can not run it at every parameter combination of interest, which limits the amount of information we have for emulation.

(2) The LAMMPS model is stochastic in nature - this introduces much randomness in the data.

(3) The model produces high-dimensional and multiple outputs which make the emulation more computationally demanding than usual.

(4) The LAMMPS model is dynamic - involves a sequence of outputs at different time points.

Despite all these caveats, the good news is that there is a large knowledge base addressing these problems. There is a limited number of literature that treat emulation of stochastic simulators. Earlier work of Kleijnen & Beers (2005) performs ordinary kriging emulation of detrended and standardized response $\mathbf{y}'$ from stochastic outputs. The scale response is derived by repeating the simulation k times at each design point such that $\mathbf{y}' = \frac{\bar{\mathbf{y}} - \hat{f}}{\sigma^2/\sqrt{k}}$, where $\bar{\mathbf{y}}(x_i) = \frac{\sum_{j=1}^{k} \mathbf{y}_{ij}}{k}$, $\sigma^2(x_i) = \frac{\sum_{j=1}^{k} (\mathbf{y}_{ij} - \bar{\mathbf{y}})^2}{k-1}$ and $\hat{f}$ is estimate of main signal function. This approach was extended by Bates et al. (2006) where an independent GP emulator is developed for both the mean response and stochastic (noise) variance. A related approach was documented in Kersting et al. (2007) and Bates et al. (1997) where an additional GP model is built to estimate the noise variance of the noise-free dataset. In addition to developing a model for the mean response, Boukouvalas et al. (2009) also fit empirical log-transformed noise variance in an heteroscedastic GP modelling of a stochastic simulator. Similar to Bates et al. (2006); Boukouvalas et al. (2009), Henderson et al. (2012) focuses on the emulation and calibration of a stochastic computer model, implementing two independent Gaussian processes on the sample mean and log-transformed standard deviation of simulation outputs. The independent GP models use a nugget parameters to account for sampling error in the data.

Our initial treatment of the stochasticity in the model is to perform multiple runs and average the key outputs which are then taken as deterministic in nature. The second approach is to incorporate nugget terms in the form of empirical variance derived from the sampling data.

3.4.1 Dynamic emulator

Due to the nature of output data from LAMMPS model, we consider using a dynamic emulation technique. Dynamic emulation models the evolution or trajectory of random variables over some time-steps (Conti et al., 2009). Emulation of time-series data or

physical processes that evolve with time which implies that model output at time t becomes an input to the model at time $t+1$. The model can be written as

$$\mathbf{y}_t = f(\mathbf{x}_t, \mathbf{y}_{t-1}) \tag{3.1}$$

Where $\mathbf{y}_{t-1}$ is the state vector at the previous time step for $t = 1, \ldots, T$, and $\mathbf{x}_t$ are the inputs at time t which includes the model parameters, forcing and initial conditions. There are two fundamental techniques for addressing dynamic emulation as discussed in (Conti et al., 2009). The multi-step and single-step emulations and there are about three different variants of the multi-step technique.

3.4.2 Multi-step emulation

Firstly, we can emulate a complete multi-step run of the computer model. One of the ways to proceed with this according to Conti et al. (2010) is to treat the problem as a multivariate output simulator and develop a multi-output emulator where the dimension of the output space is given as T. Closely related to this approach, is to build one single-output emulator that incorporates time as an additional input to the emulator such that $\mathbf{y}_t = f(\mathbf{x}, t)$, where the training data for building emulator consists nT data points. The limitation of this approach is that it is inefficient in practice because the dimension of the data becomes vast which introduces additional computational difficulty.

The third variant is to emulate each time step, which produces an emulator that is peculiar to a particular time step, an approach that assumes independence between the time steps. This method was used in Boukouvalas et al. (2014) but is not suitable for our present data. Here, where are interested in the temporal correlations across the time steps, and specifically for using an emulator to scale up LAMMPS model outputs from an order of O(10^6) particles to O(10^{13}) particles, i.e., for making multiple-step ahead predictions.

3.4.3 Single-step emulation

The second method is to model the simpler, single step simulator and use the emulator repeatedly to generate the full-time series of the resulting predictions up to the number of desired time points. This framework reduces the dimension of the problem. We implement both methods for emulation floc equivalent diameter. Although, the single-step emulation seems much appealing for our upscaling problem considering fact that we want to capture the complete behaviour of floc diameter over a number of time steps.

We describe the single step function emulation here. We follow a similar procedure

described in Conti et al. (2009); Bhattacharya (2007). Starting from initial run of the model at time t_0, we construct the single step emulator $\mathbf{y}_1 = f(\mathbf{x}_1, \mathbf{y}_0)$ using a GP regression in form of kriging. One of the usefulness of dynamic emulation is to make a multiple step ahead predictions using iterative technique to repeat one-step-ahead predictions until the desired number of points. We proceed sequentially, feeding back the entire output distribution from the GP model, such that at time step $t = 1$, for input $(\mathbf{x}_1, \mathbf{y}_0)$, we sample from the distribution of $f(\mathbf{y}_0, \mathbf{x}_1)$, the model output is given as

$$\tilde{\mathbf{y}}_1^{(s)} \sim N\Big(\mu^\bullet(\mathbf{x}_1, \mathbf{y}_0), \mathbf{K}^\bullet(\mathbf{x}_1, \mathbf{y}_0)\Big).$$

For the next prediction at time $t = 2$, the input data $\mathbf{x}_2$ is augmented by complete distribution $\mathbf{y}_1^{(s)}$ such that $\mathbf{X}_2 = [(\mathbf{x}_2, \tilde{\mathbf{y}}_1)]^T$, then we generate sample from the distribution of $f(\tilde{\mathbf{y}}_1^{(s)}, \mathbf{x}_2)$ and denote as $\tilde{\mathbf{y}}_2^{(s)}$, note that distribution of $\tilde{\mathbf{y}}_2^{(s)}$ is no longer normally distributed. This procedure is repeated until $T-1$ steps is reached. The construction of single-step emulator is summarized below:

(i) Subsample 300 points randomly from original 1000 points and formulate a single step emulator using equation (3.1) such that $\mathbf{y}_1 = f(\mathbf{x}, \mathbf{y}_0)$, where $\mathbf{x}$ is the new design matrix for running the LAMMPS model for the single step function, $\mathbf{x}$, as usual, include initial conditions and calibrated (constant) parameters while the corresponding output is the value of current state variable $\mathbf{y}_t$.

(ii) Perform the GP emulation in the form of kriging as described in previous chapter 2, where we use a quadratic mean and Matern covariance functions. Parameters $\theta = [\hat{\beta}, \hat{\sigma^2}, \hat{\alpha}]$ are estimated by MLE technique.

(iii) Compute the posterior distribution of $\Big(f(.)|\mathbf{y}, \hat{\theta}\Big) \sim N\Big(\mu^\bullet(x_0), \mathbf{K}^\bullet(x_0)\Big)$ where $\mu^\bullet(x)$ and $\mathbf{K}^\bullet(x)$ are defined in equations (2.6, 2.7) respectively.

(iv) Use the emulator to simulate from $\Big(f(.)|\mathbf{y}_1, \hat{\theta}\Big)$ to obtain $\tilde{\mathbf{y}}_1^{(s)}$ and then iterate the next steps for $t = 1, \ldots, T-1$ to give a full time series $\Big[\tilde{\mathbf{y}}_1^{(s)}, \ldots, \tilde{\mathbf{y}}_{T-1}^{(s)}\Big]$.

(v) Derive a new training data by augmenting the original data with simulated time series and rebuild the single-step emulator with the new training data given below.

$$\begin{pmatrix} \text{Original inputs} \\ \vdots \\ (\mathbf{y}_0, \mathbf{x}_1) \\ (\tilde{\mathbf{y}}_1, \mathbf{x}_2) \\ \vdots \\ (\tilde{\mathbf{y}}_{T-1}, \mathbf{x}_T) \end{pmatrix} = \begin{pmatrix} \text{Original outputs} \\ \vdots \\ \tilde{\mathbf{y}}_1^{(s)} \\ \tilde{\mathbf{y}}_2^{(s)} \\ \vdots \\ \tilde{\mathbf{y}}_T^{(s)} \end{pmatrix}.$$

(vi) Simulate $\tilde{\mathbf{y}}_{t+1}^{(s)}$ from conditional distribution $\Big(f(.)|\mathbf{y}_t, \hat{\theta}\Big)$.

(vii) Repeat the entire process many times to obtain $\tilde{\mathbf{Y}}^N = \Big[\tilde{\mathbf{y}}_1^{(s)}, \ldots, \tilde{\mathbf{y}}_{T-1}^{(s)}\Big]^N$, for $s = 1, \ldots, N$, where $\tilde{\mathbf{Y}}^N$ is a sample from the joint distribution of $[\mathbf{y}_1, \ldots, \mathbf{y}_{T-1}]$ given the emulator training data and initial conditions and N is the number of Monte Carlo (MC) sample.

Normal approximations to recursive iterations

One of the limitations of above procedure is that it is highly prone to numerical problems associated with the inversion of the covariance matrix as training data is augmented. Moreover, an additional computational cost is often involved. There is a simple normal approximation to the above procedure that can prevent repeated use of single emulator thus avoiding the computational difficulties. Here, we summarize the approximation technique according to Conti et al. (2009). This new procedue is based on the assumption that augmentation of training data at each iteration step will have a relatively minimal effect provided that we use a large sample size for building our single-step emulator, in other words, additional data at each step could be discarded. In addition, since our training data for the single step emulator $\mathbf{y}_t = f(\mathbf{x}, t)$ is modelled as a GP, thus makes it difficult to derive a joint distribution for $\mathbf{y}_1, \ldots, \mathbf{y}_T$ in a closed form, rather a normal approximation is proposed to estimate the marginal distribution of each $\mathbf{y}_t$ for $t = 1, \ldots, T$. Suppose, the marginal distribution of $\mathbf{y}_t$ can be approximated as $\mathbf{y}_t \sim N\Big(\mu_t(.), \mathbf{K}_t(.)\Big)$, where $\mu_1 = \mu^\bullet(x_1, y_0)$ and $\mathbf{K}_1 = \mathbf{K}^\bullet\Big((x_1, y_0), (x_1, y_0)\Big)$, $\mu^\bullet(.)$ and $\mathbf{K}^\bullet(.,.)$ are already defined in equations (2.6 and 2.7) respecctively.

Then we have

$$\mu_{t+1} = E\Big(\mu^\bullet(\mathbf{y}_t, x_{t+1})|f(\mathbf{y})\Big), \tag{3.2}$$

$$\mathbf{K}_{t+1} = E\Big(\mathbf{K}^\bullet(x_{t+1}, \mathbf{y}_t), (x_{t+1}, \mathbf{y}_t)|f(\mathbf{y})\Big) + var\Big(\mu^\bullet(\mathbf{y}_t, x_{t+1})|f(\mathbf{y})\Big). \tag{3.3}$$

Now, we can then estimate the two quantities above using simulation from Monte Carlo sampling to repeatedly revise the mean and variance of the single step emulator.

$$\hat{\mu}_{t+1} = \frac{1}{N}\sum_{s=1}^{N}\Big(\mu^\bullet(\tilde{\mathbf{y}}_t^{(s)}, x_{t+1})|f(\mathbf{y})\Big), \tag{3.4}$$

$$\widehat{\mathbf{K}}_{t+1} = \frac{1}{N}\sum_{s=1}^{N}\Big(\mathbf{K}^\bullet(x_{t+1}, \tilde{\mathbf{y}}_t^{(s)}), (x_{t+1}, y_t)|f(\mathbf{y})\Big) + \frac{1}{N}\sum_{s=1}^{N}\Big(\mu^\bullet(\tilde{\mathbf{y}}_t^{(s)}, x_{t+1})|f(\mathbf{y})\Big)^2, \tag{3.5}$$

where $\tilde{\mathbf{y}}_t^{(s)}$ is a sample from $N\Big(\mu_t(.), \mathbf{K}_t(.)\Big)$. This approximation technique is also related to procedure earlier described in Azman & Kocijan (2005); Girard et al. (2003); Azman & Kocijan (2005b) where GP is applied to a nonlinear dynamic systems to

propagate uncertainty in an iterative multiple-step-ahead predictions.

3.5 Results

Suppose at time step t, the LAMMPS output is written in the form

$$\mathbf{y}_t = f(\mathbf{x}_t, \mathbf{y}_{t-1}) \tag{3.6}$$

Where $\mathbf{y}_{t-1}$ the state vector at the previous time step, $\mathbf{x}_t$ are the input at time t which includes the model parameters, forcing and initial conditions as described earlier. We summarize the indiviudual particle at microscale to a large (mesoscale) as a floc. We consider emulation of floc which is summarized by aggregating all the individual microbe at each time step. The number of particles n at each time slice varies across the design points as stated before. The number of design points at each time step is 1000 and $T = 86$ in our simulation. The total floc mass at time t is given as

$$M_t = \sum_{k=1}^{n} m_{kt}, \tag{3.7}$$

and center of mass $\tilde{C}$ for the floc aggregate in 3-dimension, for X direction using equation

$$\tilde{P}_{x_t} = \frac{\sum_{k=1}^{n} m_{kt} X_{kt}}{M_t}, \tag{3.8}$$

where M_t is the total floc mass at time t for all the species and m_{kt}'s are individual particle level mass. Replace X_{kt} in equation (3.8) with Y_{kt} and X_{kt} respectively to derive for other directions.

There are two different ways to derive the floc equivalent diameter namely the volume and distance techniques. Under the distance approach, the diameter of the smallest circle that circumscribes the outer edge or sketch of the floc can be obtained by computing relative distances in $X-Y-Z$ positions of each of the particle from the center of mass of the floc aggregate. The sum of the maximum of this distances and radius of the particle with the largest distance will form the radius of the outer sphere as shown in Figure (3.1).

Suppose at time t, the distance in euclidean three-space between any two positions, say particle p at position $P = (x_k, y_k, z_k)$ and floc center of mass at point $\tilde{P} = (x_0, y_0, z_0)$ is given as $d_k = \sqrt{(x_{kt} - x_0)^2 + (y_{kt} - y_0)^2 + (z_{kt} - z_0)^2}$,

$$d_{eqv} = 2(\max(d_k) + r_{k'}), \tag{3.9}$$

where $r_{k'}$ is the radius of particle with largest distance and x, y and z are respective

directions, $k = 1, \ldots, n$. The second approach is to compute the total volume of the floc using the volume of each individual particle (particle is taken as a sphere).

$$d_{eqv} = \sum_{k=1}^{n} \sqrt[3]{\frac{6V_{kt}}{\pi}} \tag{3.10}$$

where V_{kt} volume of individual spherical particle k at time t, π is a constant and d_{eqv} is the floc equivalent diameter. The volume technique under-estimates the value of equivalent diameter.

We apply the kriging model to train the data for floc equivalent diameter d_{eqv} using a normal approximation technique of dynamic emulation. Some results are given below.

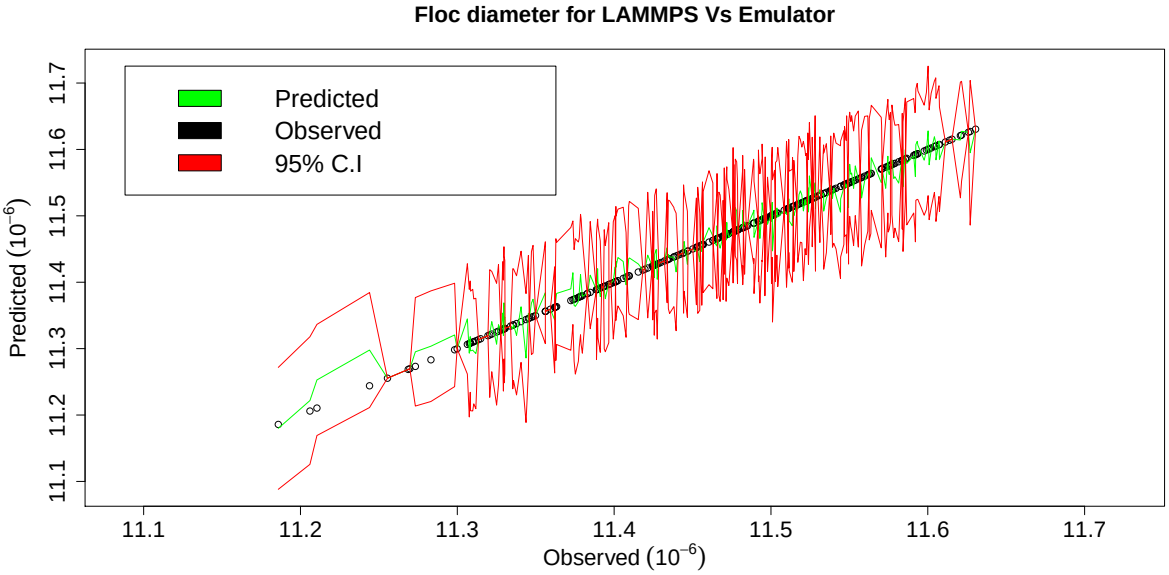


Figure 3.2: Comparison between floc diameter for LAMMPS model and emulator with 95% C.I

3.5.1 Emulator Performance

(i) It takes LAMMPS model between (13-24 hours) to obtain 2 days simulation ensembles on 8G ram, 4-cores Linux machine.

(ii) Emulator gives results almost instantaneously (2 minute).

(iii) This is 1100-fold increase in computational efficiency.

(i) Biofilm /floc total mass at each time step

(ii) Biofilm /floc equivalent diameter at each time step

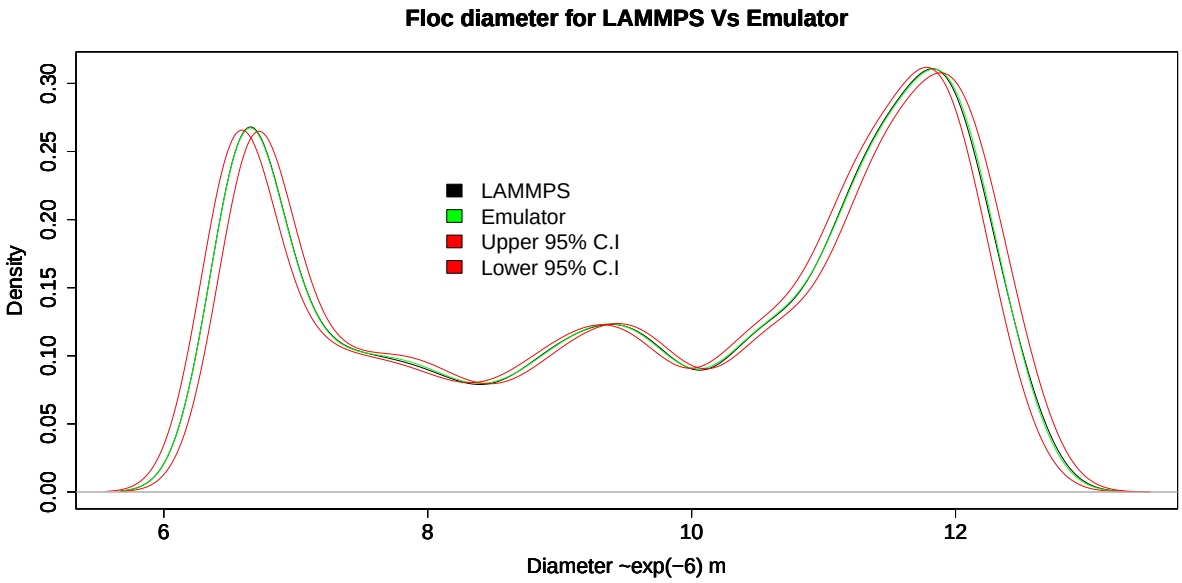


Figure 3.3: Probability density function for LAMMPS model and emulator with 95% C.I

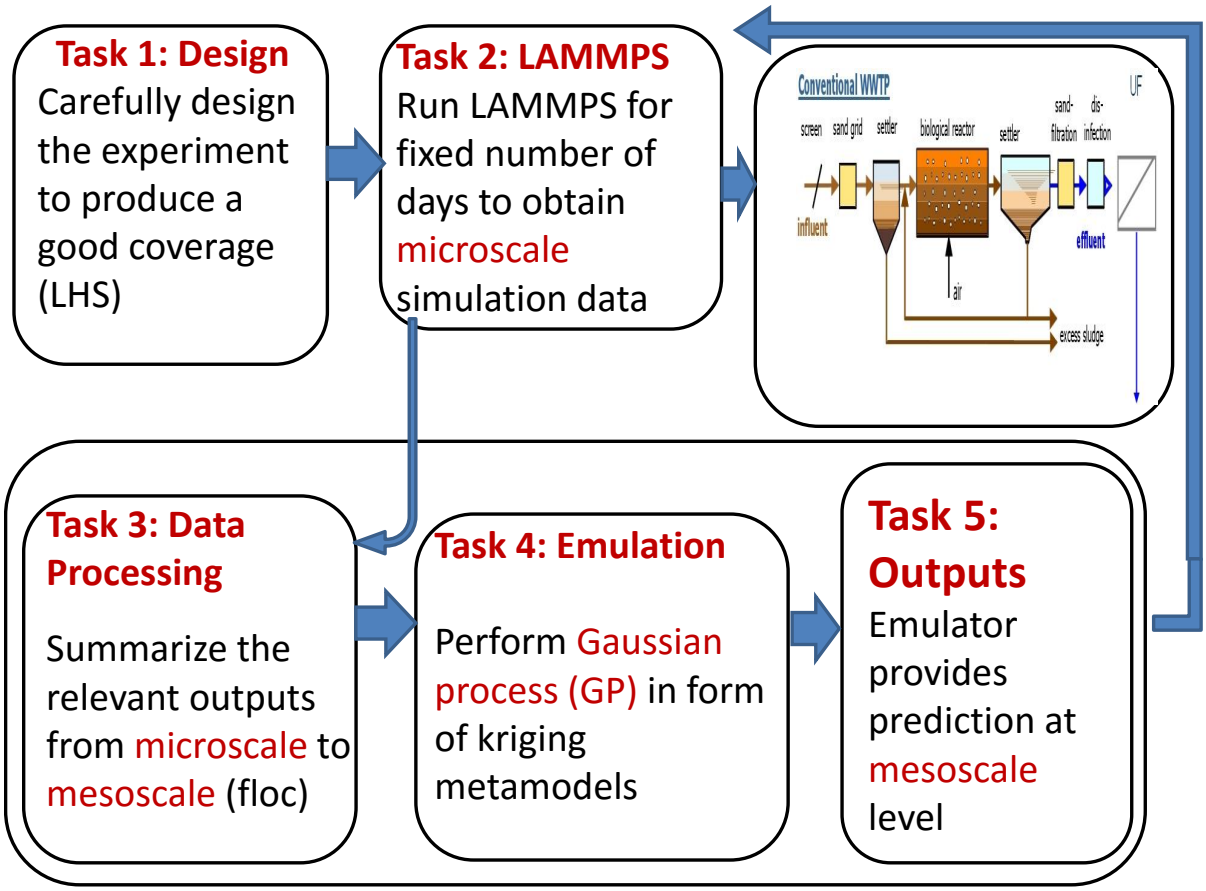


Figure 3.4: Schematic diagram showing key emulation stages

(iii) EPS total mass at each time step

(iv) Total number of particles at each time step

(v) The mass ratio of individual particle to the total biofilm /floc mass

(vi) Distribution of floc/ biofilm diameter

Chapter 4

Biofilm emulation

We describe emulation of biofilm in this section, we characterize the biofilm as shown in the Figure 4.1 below. We apply the same procedure for emulating floc to the biofilm modelling.

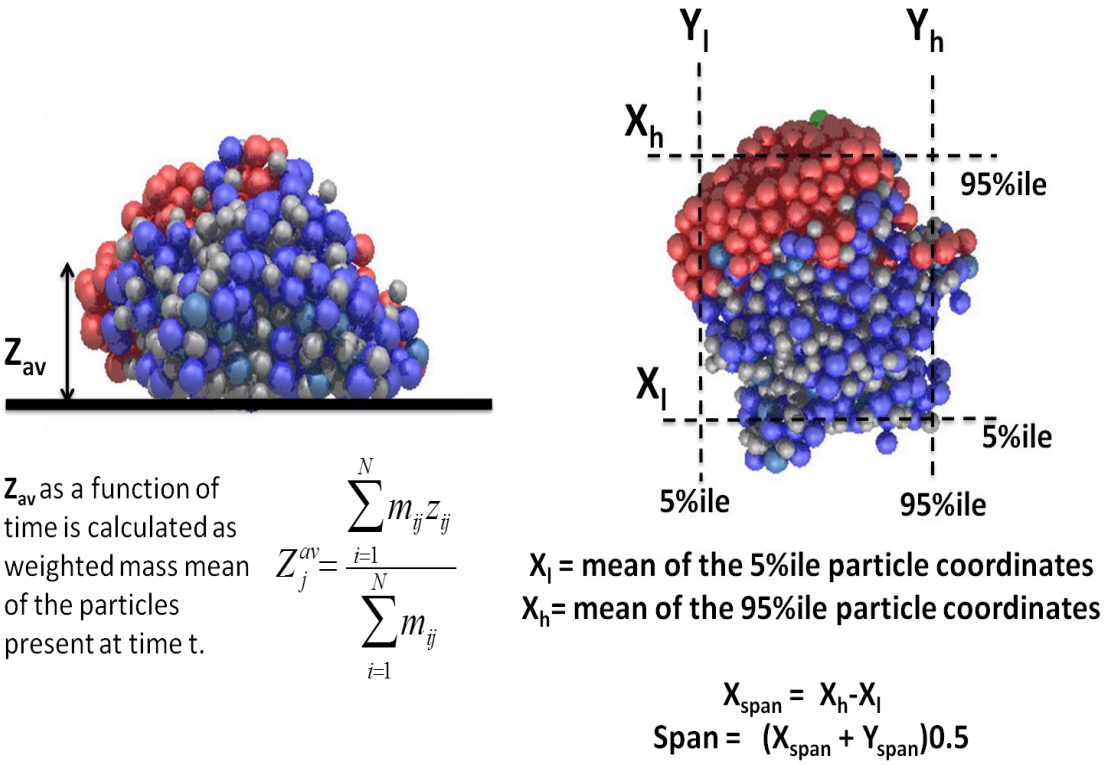


Figure 4.1: Biofilm characterization

Bibliography

Currin, C., Mitchell, T.J., Morris, M.D., and Ylvisaker, D. (1991). Bayesian Prediction of Deterministic Functions, With Applications to the Design and Analysis of Computer Experiments. *Journal of the American Statistical Association*, 86(416), 953 – 963. 8

Martin, J. D., & Simpson, T. W. (2004). On the use of kriging models to approximate deterministic computer models. *In ASME 2004 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, 481 – 492. 8

Osio, I.G. and Amon, C.H. (1996). An Engineering Design Methodology with Multistage Bayesian Surrogate and Optimal Sampling. *Research in Engineering Design*, 8(4), 189 – 206. 8

Sacks, J., Welch, W., Mitchell, T., Wynn, H. (1998). Design and analysis of computer experiments. *Statistical Science*, 4(4), 409 – 435. 9

Santner, T., Williams, B., Notz, W. (2003). The Design and Analysis of Computer Experiments. Springer. 1, 9, 28

Li, R., & Sudjianto, A. (2005). Analysis of computer experiments using penalized likelihood in gaussian kriging models. *Technometrics*, 47(2), 111 – 120. 8

Andrianakis, Y., & Challenor, P. G. (2009). Parameter estimation and prediction using gaussian processes. *Technical report*, MUCM Technical report 09/05, University of Southampton. 5, 28

Roustant, O., Ginsbourger, D., & Deville, Y. (2012). DiceKriging, DiceOptim: Two R packages for the analysis of computer experiments by kriging-based metamodeling and optimization. 10

Park J.S., & Baek, J. (2001). Efficient Computation of Maximum Likelihood Estimators in a Spatial Linear Model with Power Exponential Covariogram. *Computer Geosciences*, 27, 1 – 7. 10

Hankin, R. K. (2005). Introducing BACCO, an R package for Bayesian analysis of computer code output. *Journal of Statistical Software*, 14, 16. 5

O'Hagan, A. (2006). Bayesian Analysis of Computer Code Outputs: A Tutorial. *Reliability Engineering and System Safety*, 91, 1290 – 1300. 3

Conti, S., Gosling, J. P., Oakley, J. E., & O'hagan, A. (2009). Gaussian process emulation of dynamic computer codes. *Biometrika*, asp028. 16, 17, 18, 19

Conti, S., Anderson, C. W., Kennedy, M. C., & OHagan, A. (2004). A Bayesian analysis of complex dynamic computer models. *In Proc. of the 4th International Conference on Sensitivity Analysis of Model Output*.

Conti, S., & OHagan, A. (2010). Bayesian emulation of complex multi-output and dynamic computer models. *Journal of statistical planning and inference*, 140(3), 640 – 651. 17

Bhattacharya, S. (2007). A simulation approach to Bayesian emulation of complex dynamic computer models. *Bayesian Analysis*, 2(4), 783 – 815. 18

Azman, K., & Kocijan, J. (2005). Comprising prior knowledge in dynamic gaussian process models. *In Proceedings of the International Conference on Computer Systems and Technologies-CompSysTech*, Vol. 16(17.6). 19

Kleijnen, J. P. (2009). Kriging metamodeling in simulation: A review. *European Journal of Operational Research*, 192(3), 707 – 716. 9

Martin, J. D., & Simpson, T. W. (2005). Use of kriging models to approximate deterministic computer models. *AIAA journal*, 43(4), 853 – 863. 9

Kleijnen, J. P., & Mehdad, E. (2014). Multivariate versus univariate kriging metamodels for multi-response simulation models. *European Journal of Operational Research*, 236(2), 573 – 582. 9

Kleijnen, J. P. (2009) Boukouvalas, A., Cornford, D., & Singer, A. (2009). Managing uncertainty in complex stochastic models: *Design and emulation of a rabies model. In 6th St. Petersburg Workshop on Simulation*, (pp. 839-841). 16

Kersting, K., Plagemann, C., Pfaff, P., & Burgard, W. (2007). Most likely heteroscedastic Gaussian process regression. *In Proceedings of the 24th international conference on Machine learning*, (pp. 393-400). ACM. 16

Kleijnen, J.P., & Van Beers, W.C. (2005). Robustness of kriging when interpolating in random simulation with heterogeneous variances: Some experiments. *European Journal of Operational Research*, 165(3), 826 – 834. 16

Bates, R. A., Kenett, R. S., Steinberg, D. M., & Wynn, H. P. (2006). Achieving robust design from computer simulations. *Quality Technology and Quantitative Management*, 3(2), 161 – 177. 16

Goldberg, P. W., Williams, C. K., & Bishop, C. M. (1997). Regression with input-dependent noise: A Gaussian process treatment. *Advances in neural information processing systems*, 10, 493 – 499. 16

Henderson, D. A., Boys, R. J., Krishnan, K. J., Lawless, C., & Wilkinson, D. J. (2012). Bayesian emulation and calibration of a stochastic computer model of mitochondrial DNA deletions in substantia nigra neurons. *Journal of the American Statistical Association*. 16

Boukouvalas, A., Sykes, P., Cornford, D., & Maruri-Aguilar, H. (2014). Bayesian precalibration of a large stochastic microsimulation model. *Intelligent Transportation Systems, IEEE Transactions on*, 15(3), 1337 – 1347. 17

Kleijnen, J., & Mehdad, E. (2012). Kriging in multi-response simulation, including a Monte Carlo laboratory. CentER Discussion Papers Series, (2012-039). 9

Girard, A., Rasmussen, C. E., Quinonero-Candela, J.,& Murray-Smith, R. (2003). Gaussian Process Priors With Uncertain Inputs: Application to Multiple-Step Ahead Time Series Forecasting. *In Becker, S., Thrun, S., and Obermayer, K., editors, Advances in Neural Information Processing Systems 15*. MIT Press. 19

Azman, K., & Kocijan, J. (2005). An example of Gaussian process model identification. *In Proc. 28th International conference MIPROCIS (Eds.: Budin, L. and Ribaric S.)*, Opatija (pp. 79-84). 19

Jarvis, P., Jefferson, B., & Parsons, S. A. (2005). Measuring floc structural characteristics. *Reviews in Environmental Science and Bio/Technology*, 4(1 – 2), 1 – 18. 14

Appendix 1: Estimation of prior hyperparameters

Noting that under GP regression, the prior distribution for the data is also a Gaussian distribution, the joint likelihood of the parameters is given as

$$p(\beta, \sigma^2, \alpha|\mathbf{y}) \propto \frac{det(\mathbf{C})^{-\frac{\mathbf{1}}{\mathbf{2}}}}{(2\pi\sigma^2)^{\frac{n}{2}}} \exp\left\{\frac{(y - H\beta)^T \mathbf{C^{-1}} (\mathbf{y} - \mathbf{H}\beta)}{2\sigma^2}\right\} \tag{A.10}$$

where $\alpha = [\alpha_1, \ldots, \alpha_n]$ is a vector of correlation lengths and $det(\mathbf{C})$ is the determinant of correlation matrix $\mathbf{C}$, integrating out β using a non-informative (uniform) prior such that $p(\beta) \propto 1$. We have a marginal likelihood

$$p(\mathbf{y}|\sigma^2, \alpha) \propto \frac{\det(\mathbf{C})^{-\frac{1}{2}} \det(\mathbf{H^T C H})^{-\frac{1}{2}}}{(2\pi\sigma^2)^{\frac{n-p}{2}}} \exp\left\{\frac{(\mathbf{y} - \mathbf{H}\hat{\beta})^T \mathbf{C}^{-1} (\mathbf{y} - \mathbf{H}\hat{\beta})}{2\sigma^2}\right\}. \quad \text{(A.10b)}$$

Maximizing (A.10b) with respect to β and σ^2 will give

$$\hat{\beta} = (\mathbf{H}^T \mathbf{C}^{-1} \mathbf{H})^{-1} \mathbf{H}^T \mathbf{C}^{-1} \mathbf{y} \quad \text{(A.11)}$$

$$\widehat{\sigma^2} = \frac{1}{n-p}\left[(\mathbf{y} - \mathbf{H}\hat{\beta})^T \mathbf{C^{-1}} (\mathbf{y} - \mathbf{H}\hat{\beta})\right] \quad \text{(A.12)}$$

Now, integrate out σ^2 such that $p(\sigma^2) \propto \frac{1}{\sigma^2}$, then we have

$$p(\mathbf{y}|\alpha) \propto \det(\mathbf{C})^{-\frac{1}{2}} \det(\mathbf{H}^T \mathbf{C H})^{-\frac{1}{2}} \left\{(\mathbf{y} - \mathbf{H}\hat{\beta})^T \mathbf{C}^{-1} (\mathbf{y} - \mathbf{H}\hat{\beta})\right\}^{-(n-p)/2} \quad \text{(A.10c)}$$

$$= \det(\mathbf{C})^{-\frac{1}{2}} \det(\mathbf{H}^T \mathbf{C H})^{-\frac{1}{2}} \widehat{\sigma^2}^{-(n-p)/2}.$$

The smoothing parameter α is estimated from the posterior distribution using the posterior mode by the value of α for which marginal likelihood (A.10c) is maximised. Here, we describe briefly the maximisation of the posterior distribution of α which is reparametrized as $\tau = 2log(\alpha)$ to make it unconstrained optimisation (Andrianakis & Challenor, 2009; Santner et al., 2003). Therefore, function $f(\tau) = log(p(\mathbf{y}|\exp(\frac{\tau}{2})))$ can be optimised using a derivative-free numerical optimisation of Nelder-Mead method which is the default in the "emulator" package that we use.

Table 4.1: List of LAMMPS model parameters

Index	List of parameters	Value
1	KsHET	0.01
2	Ko2HET	0.81
3	Kno2HET	0.0003
4	Kno3HET	0.0003
5	Knh4AOB	0.001
6	Ko2AOB	0.0005
7	Kno2NOB	0.0013
8	Ko2NOB	0.00068
	Defining maximum growth variables	
9	MumHET	0.00006944444
10	MumAOB	0.00003472222
11	MumNOB	0.00003472222
12	etaHET	0.6
	Defining decay rates variables	
13	bHET	0.00000462962
14	bAOB	0.00000127314
15	bNOB	0.00000127314
16	bEPS	0.00000196759
17	YEPS	0.18
18	YHET	0.61
19	EPSratio	1.25
20	factor	1.5
	Initial conditions (nutrients)	
21	sub	0.08
22	no2	0.008
23	no3	1e-05
24	o2	0.01
25	nh4	0.09