Efficient meta-modelling of complex process simulations with time-space-dependent outputs

Tao Chen[*,1], Kunn Hadinoto[1], Wenjin Yan[1], Yifei Ma[2]

[1] *School of Chemical and Biomedical Engineering, Nanyang Technological University, 62 Nanyang Drive, Singapore 637459, Singapore*

[2] *Defence Medical and Environmental Research Institute, DSO National Laboratories, 20 Science Park Drive, Singapore 118230, Singapore*

Abstract

Process simulations can become computationally too complex to be useful for model-based analysis and design purposes. Meta-modelling is an efficient technique to develop a surrogate model using "computer data", which are collected from a small number of simulation runs. This paper considers meta-modelling with time-space-dependent outputs in order to investigate the dynamic/distributed behaviour of the process. The conventional method of treating temporal/spatial coordinates as model inputs results in dramatic increase of modelling data and is computationally inefficient. This paper applies principal component analysis to reduce the dimension of time-space-dependent output variables whilst retaining the essential information, prior to developing meta-models. Gaussian process regression (also termed kriging model) is adopted for meta-modelling, for its superior prediction accuracy when compared with more traditional neural networks. The proposed methodology is successfully validated on a computational fluid dynamic simulation of an aerosol dispersion process, which is potentially applicable to industrial and environmental safety assessment.

Key words: Computer experiments; Design of Experiments; Gaussian process; Kriging model; Meta-model; Principal component analysis.

1. Introduction

In the field of systems approach to process engineering, the development of mathematical models plays a paramount role to achieve various goals ranging from process understanding, off-line optimal design, on-line real-time optimization to process control. A notable trend in process systems engineering is the ever-increasing model complexity, which may be defined as the amount of computation required to solve the model. In chemical engineering, complex models mainly originate from the physical scales being considered. For example, a complex plant-wide model (i.e. flowsheet simulation) is typically implemented by combining the models for individual processing units. Another example is that a simple reactor model based on ordinary differential equations (ODEs) becomes more complex if the spatial variation within the reactor is not negligible, and thus partial differential equations (PDEs) have to be applied. Process models are even more demanding in terms of computation if meso- and micro-scale phenomena are considered, such as computational fluid dynamic (CFD) models and molecular simulations. In general, complex models are capable of representing the underlying process more realistically and accurately. Therefore, the computational cost is among the major obstacles for the wide acceptance of complex models in practice.

To address the computational challenge, several techniques have been proposed in the literature. The method of "model reduction" is primarily designed to reduce the number of ODEs, which are typically the result of discretizing PDEs, using principal component analysis (PCA) (Gay & Ray, 1995; Hoo & Zhang, 2001) and approximate inertial manifolds (Shvartsman et al., 2000). As indicated by Romijn et al. (2008), purely reducing the number of equations does not automatically reduce computation, since the complexity in evaluating the non-linear equations is intact. Following this argument, Romijn et al. (2008) combined PCA with a grey-box approach, whereby the non-linear part of the ODEs is approximated by an empirical neural network (NN) model. The resulting reduced model runs sufficiently fast for real-time applications, such as model-based predictive and optimizing control.

[*] Corresponding author. E-mail: chentao@ntu.edu.sg; Tel.: +65 6513 8267; Fax: +65 6794 7553.

As opposed to on-line applications, an alternative category of techniques are originally targeted at off-line process understanding and design. Early work in this category was presented in the community of applied statistics (Kennedy & O'Hagan, 2001; Sacks et al., 1989). The basic concept is to treat the simulation as "computer experiments" (as opposed to physical/chemical experiments), and then apply the methodology of design of experiments (DoE) and response surface methodology (RSM) to study the impact of process inputs (e.g. operating conditions) on outputs (e.g. process yield). Similar to design and analysis of real experiments (Myers & Montgomery, 1995), the method is to properly design the value of inputs, at which the complex model is simulated to obtain the output. Then, the simulated "data" (input-output pairs) are used to develop an empirical model (termed meta-model or surrogate model), which can be used in place of the original complex model for process analysis and design. Since the empirical model runs much faster than the original complex model, the computational cost is mainly determined by the number of original model runs, which can be decided based on available resources. Compared with the grey-box model reduction technique, meta-modelling is a black-box approach and is especially suitable to be used with third-party simulation tools, such as commercial flowsheet software and CFD tools. Recently, meta-modelling has been introduced into process systems engineering for the optimization of a PDE model for radiant-convective drying (Dutournie, et al., 2006) and flowsheet simulations (Caballero & Grossmann, 2008a,b; Palmer & Realff, 2002). Gomes et al. (2008) also demonstrated the extension of meta-modelling for real-time optimization.

The above reviewed meta-modelling studies are exclusively dealing with a univariate output variable, which typically measures the process performance (e.g. process yield or dollar-value profit). However, meta-modelling to relate multivariate output with process inputs is required in some situations, where the output variable is a function of time-space coordinates. This is especially the case when the model is used beyond optimization purposes. For example, in a CFD simulation of a large-scale reactor, the reactant concentration field as a function of stirring speed may be of interest to visualize the efficiency of the stirrer. Another example may be the CFD modelling of hazardous gas dispersion, where the entire concentration field of contaminants needs to be presented for risk assessment. In these situations, the dimension of the output variable is dependent on the number of temporal and spatial steps to solve the PDEs, and it can easily exceed the order of thousands or even millions. A usual method is to treat the time-space coordinates as additional input variables, and thus the output becomes univariate (Kennedy & O'Hagan, 2001; Zhang et al., 1998). However, this method results in a very large number of data points (equal to the number of model runs times the number of time-space coordinates), posing significant difficulty for meta-modelling. Recently in the statistical community, a rigorous method was developed to model multiple outputs simultaneously (Rougier, 2008), which is restricted to several hundred dimensions due to computation. An alternative method is to use wavelet basis function to efficiently represent the time-dependent output (Bayarri et al., 2007), whereas the spatial dependency was not considered.

The major contribution of this paper is to extend the meta-modelling method with time-space-dependent process outputs. The main component is to apply PCA to reduce the high-dimensional output to low-dimensional score vector, and then develop meta-models to predict each score. Then, the original output can be re-constructed from the predicted score vector. The rationale of using PCA is that the output variables at nearby tempo-spatial coordinates are highly correlated (Gay & Ray, 1995; Rougier, 2008), and thus can be efficiently represented by a low-dimensional score vector. In addition, by using separate meta-models for individual scores, we essentially assume that these score variables are independent. This appears to a reasonable assumption, since the scores are uncorrelated due to the application of PCA (Jolliffe, 2002).

A specific empirical modelling technique, the Gaussian process (GP) also known as kriging model, is chosen as the meta-model in this study. Compared with polynomial functions in conventional RSM and more advanced NN, GP has been recognized to attain both accurate prediction and the capability of quantifying its own prediction uncertainty. The latter property is especially important, since the mismatch between meta-model and original complex model needs to be quantified to give reliable and meaningful

results (Caballero & Grossmann, 2008b; Kennedy & O'Hagan, 2001; O'Hagan, 2006; Palmer & Realff, 2002). In addition, GP model also provides the mechanism to quantify the mismatch between the complex model and the real process, if experimental data are available (Higdon et al, 2008; Kennedy & O'Hagan, 2001), though this topic is beyond the scope of this study.

The rest of this paper is organized as follows. Section 2 briefly introduces the DoE method to determine proper values for process inputs to run the complex model. Section 3 gives an overview of GP regression, and its extension to meta-modelling of time-space-dependent output variable. Section 4 demonstrates the proposed method on a CFD simulation of an aerosol dispersion process, which is a simplified scenario to study the malicious release of hazardous materials within a confined space. The example is used to illustrate that meta-models are capable of predicting the two-dimensional concentration field of aerosols across time at different inlet aerosol concentration, particle density and air velocity, whereby the output dimension under investigation is up to 182,250. The prediction accuracy of GP meta-models is compared with that of NN. Finally Section 5 concludes this paper.

2. Design of computer experiments

In the context of computer experiments, the key objective of DoE is to select the values of simulation inputs in such a way that the obtained simulation data are representative of the input space being explored and informative to predict the process outputs. Note that in computer experiments, the simulation inputs may include both process factors (e.g. inlet flow rate) and model parameters (e.g. specific heat). In situations where the model parameters cannot be obtained exactly, the impact of these parameters on simulation output must also be investigated. The classical DoE methods for real experiments, e.g. factorial designs (Myers & Montgomery, 1995), typically assign two or three pre-determined levels for each process input, and then conduct experiments at the combinations of the levels of different inputs. Using a small number of levels is favourable if the inputs are difficult to change in real experiments. However, this strategy may not have an optimal coverage of the design space due to limited levels of the factors being studied, and thus it may result in a less reliable empirical model (Fang et al., 2000). The recognition of this disadvantage of classical DoEs has motivated the concept of "space-filling" designs that allocate design points to be uniformly distributed within the range of each input.

One straightforward space-filling design is to generate Monte Carlo random samples for the multivariate uniformly distributed inputs. However, the pure randomness of this method implies that a relatively large number of samples (i.e. design points) are needed to achieve a good uniformity, which translates into a large number of simulation runs. To overcome this problem, stratified and deterministic sampling methods have been proposed to provide a good coverage of the input space with minimal number of design points, such as Latin hypercube sampling (LHS) (McKay et al., 1979), uniform design (Fang et al., 2000) and Hammersley sequence sampling (HSS) (Kalagnanam & Diwekar, 1997). In this study, the HSS design is adopted, because it has been shown to attain improved uniformity over random sampling and LHS, and its implementation is significantly easier than the number-theory-based uniform design. In addition, empirical comparison demonstrated that HSS and uniform design usually achieve comparable results (Chen et al., 2006).

The HSS design is based on the fact that any integer n can be written in a radix notation of another integer R as follows (Kalagnanam & Diwekar, 1997):

$$\begin{aligned} n \equiv\ & n_0 n_1 n_2 \cdots n_{m-1} n_m \\ =\ & n_m + n_{m-1} R + n_{m-2} R^2 + \cdots + n_1 R^{m-1} + n_0 R^m \end{aligned} \quad (1)$$

where m is the integral part of $\log_R n$. A function of n, called inverse radix number, can be constructed by reversing the order of the digits of n and concatenating them behind a decimal point:

$$\begin{aligned} \phi_R(n) =\ & 0.n_m n_{m-1} \cdots n_2 n_1 n_0 \\ =\ & n_m R^{-1} + n_{m-1} R^{-2} + \cdots + n_1 R^{-m} + n_0 R^{-m-1} \end{aligned} \quad (2)$$

Suppose that the input variable $\mathbf{x}$ is a K dimensional vector, and we select the first $K-1$ prime numbers

as the integer R in eq. (1): $R_1, R_2, \cdots, R_{K-1}$. According to HSS, the N design points, each being a vector of order K, are given by

$$\mathbf{x}_n = \mathbf{1} - \left(\frac{n}{N}, \phi_{R_1}(n), \phi_{R_2}(n), \cdots, \phi_{R_{K-1}}(n)\right)^{\mathrm{T}}, \qquad n = 1,2,\cdots,N \tag{3}$$

where $\mathbf{1}$ is a unity vector.

3. Meta-modelling using Gaussian process

The choice of a specific meta-model structure is a critical decision in meta-modelling. Naturally the meta-model should (i) give sufficiently accurate approximation to the original complex model, and (ii) provide a realistic estimation of the prediction uncertainty. Whilst the first criterion has received primary attention in the literature, the second criterion is equally important. Due to the unavoidable mismatch between the metal-model and original model, reliable quantification of meta-model's prediction error is the cornerstone for further use of the meta-model (Jones, 2001; Sacks et al., 1989).

The traditional method is to fit a polynomial function (typically linear, quadratic or cubic polynomial) to the simulation data (Dutournie et al, 2006; Palmer & Realff, 2002). It has been well recognized that polynomial regression does not possess the flexibility to fit complex input-output relationship accurately (O'Hagan, 2006; Palmer & Realff, 2002). Other meta-modelling choices include neural networks (NN) and support vector regression (SVR) (Chen et al., 2006). However, empirical studies have found that these methods, when compared with GP, typically underestimate the prediction uncertainty (O'Hagan, 2006). In recent years, GP has become a desired meta-model in various applications (Caballero & Grossmann, 2008a,b; Gomes et al., 2008; Palmer & Realff, 2002; Tang et al., 2010), and it is adopted in this study. Next, we will give a brief overview of GP regression model, followed by its extension to the prediction of time-space-dependent output variables.

3.1. Gaussian process regression model

Gaussian process (GP), also termed kriging model in the literature with slightly different formulation (Jones, 2001; Sacks et al., 1989), is a flexible modelling technique that can be used for both regression and classification purposes (Rasmussen & Williams, 2006). Previous studies have shown that GP regression model is capable of modelling complex non-linear processes accurately, as well as quantifying the prediction uncertainty reliably (O'Hagan, 2006). The latter property is crucial for further use of the meta-model (e.g. process analysis and/or optimization), since the ignorance of model-reality mismatch would result in unreliable results. Recently, GP models have seen successful applications in various fields, including chemometric calibration of spectrometers (Chen & Martin, 2009), chemical process control (Likar & Kocijan, 2007), and process modelling and optimization (Chen & Ren, 2009; Tang et al., 2010; Yuan et al., 2008). In this subsection, a brief overview of GP regression technique is given, including the formulation and implementation of the model. More details about GP can be found in (Rasmussen & Williams, 2006).

From the perspective of a regression problem, a functional relationship is identified between the K dimensional process input, $\mathbf{x}$, and the scalar output y (extension to multivariate response is given in Section 3.2): $y = f(\mathbf{x})$. Consider a training data set of size N: $\{\mathbf{x}_n, y_n; n = 1,\ldots,N\}$ that was obtained by N runs of the simulation at the designed points. A GP regression model is defined such that the regression function $y(\mathbf{x})$ has a Gaussian prior distribution with zero mean, or in discrete form:

$$\mathbf{y} = (y_1,\ldots,y_N)^{\mathrm{T}} \sim G(0,\mathbf{C}) \tag{4}$$

where $\mathbf{C}$ is an $N\times N$ covariance matrix of which the ij-th element is defined by a covariance function: $C_{ij} = C(\mathbf{x}_i, \mathbf{x}_j)$. An example of such a covariance function is:

$$C(\mathbf{x}_i, \mathbf{x}_j) = a_0 + a_1 \sum_{k=1}^{K} x_{ik} x_{jk} + v_0 \exp\left(-\sum_{k=1}^{K} w_k (x_{ik} - x_{jk})^2 \right) + u\delta_{ij} \quad (5)$$

where x_{ik} is the k-th variable of $\mathbf{x}_i$, and $\delta_{ij} = 1$ if $i=j$, otherwise $\delta_{ij} = 0$. We term $\boldsymbol{\theta} = (a_0, a_1, v_0, w_1, \ldots, w_K, u)^{\mathrm{T}}$ "hyper-parameters" defining the covariance function. The hyper-parameters must be non-negative to ensure that the covariance matrix is non-negative definite. For the covariance function given in eq. (5), the first two terms represent a constant bias (offset) and a linear correlation term, respectively. The exponential term is similar to the form of a radial basis function, and it takes into account the potentially strong correlation between the responses with similar predictors. The term u captures the random error effect, which may be due to the use of stochastic optimization algorithm for solving the simulation, among other sources. By combining both linear and non-linear terms in the covariance function, GP is capable of handling both linear and non-linear data structures (Chen & Martin, 2009). Other forms of covariance functions have been discussed by Rasmussen & Williams (2006).

For a new data point $\mathbf{x}^*$, the predictive distribution of the response y^* conditional on the training data is also Gaussian, of which the mean ($\hat{y}^*$) and variance ($\sigma^2_{\hat{y}^*}$) are calculated as follows:

$$\hat{y}^* = \mathbf{k}^{\mathrm{T}}(\mathbf{x}^*)\mathbf{C}^{-1}\mathbf{y} \quad (6)$$

$$\sigma^2_{\hat{y}^*} = C(\mathbf{x}^*, \mathbf{x}^*) - \mathbf{k}^{\mathrm{T}}(\mathbf{x}^*)\mathbf{C}^{-1}\,\mathbf{k}\ (\mathbf{x}^*) \quad (7)$$

where $\mathbf{k}(\mathbf{x}^*) = [C(\mathbf{x}^*, \mathbf{x}_1), \ldots, C(\mathbf{x}^*, \mathbf{x}_N)]^{\mathrm{T}}$. The predictive uncertainty is expressed in the variance term as in eq. (7).

The hyper-parameters $\boldsymbol{\theta}$ can be estimated by maximizing the log-likelihood of the data: $L = \log p(\mathbf{y}|\boldsymbol{\theta}, \mathbf{x}_{1:N})$, which can be solved by using gradient based methods, e.g. the conjugate gradient method (Rasmussen & Williams, 2006). The maximum likelihood estimation is usually not a convex optimization problem, and thus multiple local optima may exist. To alleviate the effect of local optima, we adopted the common practice to run the optimization algorithm multiple (ten in this study) times, each time starting from randomly selected initial value of the hyper-parameters (Rasmussen & Williams, 2006). Then, the model with the largest log likelihood was used for prediction. A Matlab implementation of the GP models is publicly available from http://www.gaussianprocess.org/gpml/code/matlab/doc/, and it was used to produce the results in this study.

3.2. Meta-modelling with time-space-dependent outputs

The extension of GP model to handle time-space-dependent outputs is nontrivial. One possibility is to transform the multivariate output $\boldsymbol{z} = [y(\boldsymbol{s}_1, t_1), \ldots, y(\boldsymbol{s}_D, t_D)]$ to univariate variables by using the spatial ($\boldsymbol{s}$) and temporal (t) coordinates as inputs: $y = f(\boldsymbol{x}, \boldsymbol{s}, t)$, where D is equal to the number of time steps multiplying the number of spatial coordinates (Kennedy & O'Hagan, 2001; Zhang et al., 1998). The major issue with this method is that the number of data points increases dramatically from N (number of simulation runs) to $N \times D$. The algorithm to estimate the hyper-parameters in a GP model involves a matrix inversion step and takes time of the order $O[(N \times D)^3]$. For conventional computers, when the number of data points is more than several thousand, special "sparse estimation" strategies must be employed to reduce the computational cost (Csato & Opper, 2002). However, the "sparse GP model" will introduce additional prediction errors. In addition, in typical simulations involving hundreds of spatial coordinates and thousands of time steps, D becomes a very large number, which causes significant computational challenge even for the sparse estimation techniques.

An alternatively approach is to reduce the dimension of the output by exploiting the correlation structure of the variables. The rationale is that the process outputs at different time and space locations are strongly correlated, and thus they can be represented by a small number of latent variables that are extracted by

principal component analysis (PCA) (Jolliffe, 2002). PCA is a general multivariate statistical projection technique for dimension reduction, and it has wide applications in process data analysis (Venkatasubramanian et al., 2003), model reduction (Gay & Ray, 1995; Hoo & Zhang, 2001; Romijn et al., 2008), among other areas. The central idea of PCA is to project the original D dimensional data, $\mathbf{z}$, onto a space where the variance is maximized: $\boldsymbol{z} = \boldsymbol{W}\boldsymbol{t} + \boldsymbol{\mu} + \boldsymbol{e}$. Here $\boldsymbol{W}$ refers to the eigenvectors of the data covariance matrix corresponding to the Q ($Q \leq D$) largest eigenvalues, $\boldsymbol{t}$ is the Q-dimensional scores, $\boldsymbol{\mu}$ is the mean of the data, and $\boldsymbol{e}$ is the noise term with zero mean and covariance matrix $\boldsymbol{S}_e$. To simply computation, it is common practice to assume that $\boldsymbol{S}_e = \gamma \boldsymbol{I}$, where $\boldsymbol{I}$ is an identity matrix and γ can be efficiently estimated from the data under a probabilistic framework (Tipping & Bishop, 1999).

Following applying PCA to high-dimensional output $\mathbf{z}$, we develop Q GP models, one for the prediction of each variable t_q of the score vector $\boldsymbol{t} = \left[t_1, \dots, t_Q\right]^T$. By using separate GP models, we essentially assume that the individual variables of $\boldsymbol{t}$ are independent. This appears to a sensible assumption, since the scores are uncorrelated due to the application of PCA (Jolliffe, 2002). Then, given a new input $\mathbf{x}^*$, the predictions from GP models are still Gaussian distributed according to eqs. (6)(7):

$$t_q^* \sim G\left(\hat{t}_q^*, \sigma_q^2\right), \qquad q = 1, \dots, Q \tag{8}$$

or in multivariate form: $\boldsymbol{t}^* \sim G(\hat{\boldsymbol{t}}^*, \boldsymbol{\Sigma})$, where $\boldsymbol{\Sigma} = diag(\sigma_1^2, \dots, \sigma_Q^2)$ is a diagonal matrix. Therefore, the prediction for the original multivariate output $\boldsymbol{z}$ is also Gaussian distributed as follows:

$$\mathbf{z} \sim G\left(\boldsymbol{W}\hat{\boldsymbol{t}}^* + \boldsymbol{\mu}, \mathbf{W}\boldsymbol{\Sigma}\mathbf{W}^{\mathrm{T}} + \mathbf{S}_e\right) \tag{9}$$

The number of principal components retained should be selected so that the PCA error term $\boldsymbol{e}$ has small magnitude, and thus unnecessary prediction uncertainty due to $\boldsymbol{S}_e$ in eq. (9) can be minimized. For this reason, we suggest to select the number of components that explains at least 99% of the total variance of the data.

4. Case study

In this section, the proposed meta-modelling approach is illustrated through CFD simulation of an aerosol dispersion process in an indoor environment. The objective is to predict the concentration profile of the aerosol (output) based on three impact factors (inputs). This simulation is a simplified case to study the impact of release conditions on the spatial-temporal profile of particle concentration, which is important information for subsequent risk analysis and mitigation in the event of malicious release of such substances. Similar simulations are also widely used for industrial safety assessment (Kisa & Jelemensky, 2009), among other purposes.

4.1. The Model

The dispersion dynamics of particulate aerosols in an indoor environment is simulated using an Eulerian-Eulerian two-phase flow CFD model that is based on the mixture model of Gidaspow (1994). In the mixture model, the fluid and the particle phases are treated as interpenetrating continuum using the volume fraction as a coupling parameter. The model formulation consists of (1) the continuity equations for each phase, (2) one momentum equation for the mixture of the two phases, which includes the inter-phase drag force term that accounts for the velocity difference between the two phases, and (3) the k-ε turbulence model to describe the fluid-phase turbulence. The inter-particle collision and its impact on the turbulent fluid flow are assumed to be negligible in the mixture model, which is a reasonable assumption for the low concentration and low inertia particles investigated in the present work. The aerosol dispersion dynamics hence is governed by the fluid-phase turbulent shear flow and the particle body forces (i.e. gravitational, buoyancy).

A transient simulation is conducted for a two-dimensional space in Figure 1 using the finite element analysis in commercial CFD software COMSOL Multiphysics 3.5 (COMSOL Inc, USA). The inlet and outlet sections are open to ambient pressures. No-slip and insulated boundary conditions are employed at

the walls for the fluid and particle phases, respectively. The effects of the inlet air velocity, inlet aerosol concentration, and the aerosol particle density (solid density, i.e. mass per m^3 of aerosol particles) on the spatial and temporal aerosol concentration profiles are investigated. The diameter of the particulate aerosol is fixed at 10 μm. The inlet air velocity is varied between 0.1 and 0.3 m/s to simulate the typical air circulation velocity in an office space (Brown, 2005). The inlet aerosol concentration is varied between 1−9% and the aerosol particle density is varied between 100 and 500 kg/m^3. The simulation runs for 180 seconds, which are sufficient for the system to reach the steady-state. The raw data for each run have a spatial dimension of 4,368 and 90 time steps (sampling rate of 2 seconds), equivalent to an output vector of 393,120. The computational time for a single simulation run is approximately three minutes on a desktop computer running Windows Vista system with an Intel Core Duo 2.66GHz processor (all computation time quoted in this paper is based on this computer).

(Figure 1 around here)

4.2. Results and discussions

To conduct the simulation, HSS was used to design values for the three input variables so that the data provide a fair coverage of the input space. To mimic the practical use of a full-fledged three-dimensional CFD simulation whereby the computation would allow a very limited number of runs, we collected a total of 20 runs of simulation data. To investigate the impact of output dimension on prediction capability, we have down-sampled the original simulation data to various time-space resolution with equal intervals, as given in Table 1. The largest output dimension under study is 182,250.

(Table 1 around here)

To demonstrate the prediction performance of the proposed method, a leave-one-out cross-validation (LOOCV) strategy is adopted. LOOCV takes a single run from the entire data set as the validation data, and then develop a model using the remaining 19 runs. This procedure is repeated such that each run is used once for validation, and then the overall validation error (typically in terms of root mean squared error (RMSE)) can be calculated. The validation data appear to be scarce in terms of number of runs; however, for each run a large number of concentration values at different time-space coordinates need to be predicted. The performance of the GP-PCA model is compared with Bayesian regularized feed-forward neural network (NN-PCA), which was shown to output-perform conventional NN models (MacKay, 1992; Thodberg, 1996). Since automatic regularization is enforced, the prediction performance is relatively insensitive to the number of neurons in the hidden layer, making this tuning task significantly easier. Furthermore, the strategy of treating time-space coordinates as extra input variables, which is referred to as simply GP or NN, is also attempted for comparison. Finally, a simple memory-based regression method, the k nearest neighbours (k-NearNbr) approach, is implemented as a baseline approach. The nearest neighbours for each testing run are determined based on the Euclidean distance of inputs, which are scaled to have zero mean and unit standard deviation to avoid the impact of units. To make prediction, the outputs from nearest neighbours are weighted by the inverse distances to the test run. We tried k in the range between 1 and 10 and found that for this case, 3-NearNbr gives the best prediction. Therefore, we report the RMSEs of 1-NearNbr and 3-NearNbr. Note that k-NearNbr requires negligible computation in model development stage: it simply stores all the training data.

In addition, we use the average *negative log predictive density* (NLPD) to assess the prediction capability when prediction interval (i.e. uncertainty) is considered (Quiñonero-Candela et al., 2006). NLPD is defined via the probability of the prediction $y_{i,\mathrm{pred}}$ being equal to the actual output y_i:

$$\mathrm{NLPD} = -\frac{1}{N_p}\sum_{i=1}^{N_p} \log p(y_{i,\mathrm{pred}} = y_i) \tag{10}$$

where N_p is the total number of predictions. When the prediction is normally distributed with mean $\hat{y}_i$ and variance σ_i^2, $p(y_{i,\text{pred}} = y_i)$ corresponds to the calculation of a normal density function with mean $\hat{y}_i - y_i$ and variance σ_i^2. NLPD reaches its minimum if predictions are equal to the true value and the predictive variances are zero. It was shown (Quiñonero-Candela et al., 2006) that given a prediction, the optimal variance is the squared error of the prediction mean. Therefore, NLPD penalizes both over-confident (small variance) and under-confident (large variance) predictions, and it is a reliable criterion to quantify the prediction quality under uncertainty. The predictive variance for GP is given in eq. (7), and that for NN is obtained through the usual first-order Taylor approximation (Chryssolouris et al., 1996). NLPD will not be used to assess *k*-NearNbr, which does not provide a prediction uncertainty unless being re-formulated under a probabilistic framework (Holmes & Adams, 2002).

The overall results are summarized in Figures 2-4. Figure 2 compares the CPU time of the meta-models. By treating time-space coordinates as extra inputs, the CPU time for GP increases dramatically with the increase of output dimension. The computation is dominated by the need to maintain and invert an $(ND \times ND)$ matrix. When output dimension reaches 345 (D=345), the memory in our computer was not sufficient to build such a GP model. NN models require less computing resource than GP; yet when output dimension exceeds 2700, the computer also runs out of memory. In contrast, PCA captures the majority of the information in the high-dimensional output variable. In this study, a maximum of eight principal components are required to explain 99% of the variance in the output data, even when the output dimension goes up to 182,250. Following the application of PCA, the data set contains only 19 points (19 runs for model development in LOOCV), each associated with up to eight principal components as output. The PCA-based meta-modelling is highly efficient in terms of computation, as indicated in Figure 2. When a small number of data are available, the computation for inverting the covariance matrix in GP is negligible, and the CPU time is mainly spent on optimizing the model parameters. The slightly higher computation of NN-PCA than GP-PCA may be attributed to the fact that a neural network has more parameters than a Gaussian process. More importantly, the CPU time for both GP-PCA and NN-PCA does not vary significantly with the increase of output dimension, providing an opportunity to meta-model a highly complex process simulation.

(Figure 2, Figure 3 and Figure 4 around here)

Figures 3 and 4 give the prediction performance of various meta-models with different output dimensions. The prediction error of the simple *k*-NearNbr method is significantly higher than NN and GP. It is interesting to observe that the RMSE of *k*-NearNbr is very similar across output dimensions. In fact, the difference in output dimensions does not change the input values, nor does it affect the selected nearest neighbours. As a result, the small variation of RMSE originates from the differences when down-sampling the simulation data to different time-space resolutions.

RMSE and NLPD jointly indicate that GP-PCA consistently attains more accurate prediction than NN-PCA in terms of lower error and more reasonable quantification of the prediction uncertainty. The criterion of NLPD appears to be unconventional; yet it plays an essential role if the meta-model is used to assess the risk due to aerosol release (among other applications), whereby the prediction uncertainty is an inherent component of probabilistic risk assessment methodologies. In addition, PCA-based meta-models give similar results as the strategy of using coordinates as inputs, suggesting that PCA has successfully condensed the high-dimensional output variables with a few principal components. The worst RMSE from GP-PCA is 0.003 (corresponding to output dimension of 81,000 and 182,250), which is less than 10% of the average concentration in the space. According to our experience dealing with various complex simulations, this magnitude of accuracy should satisfy the requirement of a wide range of applications. In addition, the prediction accuracy may be further improved by collecting more computer data from simulation, if this is within the limit of available computing resource.

Recall that the PCA-based meta-models predict the concentration field indirectly by predicting the

principal components. It is interesting to notice that the prediction accuracy of these components varies significantly. As an example, Table 2 displays the RMSE of GP-PCA for the eight principal components. (The output dimension setting corresponds to Case No. 7 in Table 1.) The general trend is that the more important components (the first several) are more accurately predicted than the others. This phenomenon is desirable, since the first several components largely determine the overall accuracy of the meta-models.

(Table 2 around here)

We further utilize Case No. 7 (Table 1) to illustrate the predicted concentration field at different time instances in Figure 5. To assess how close the current system is to the steady-state, an empirical measure is used: $D = ||\mathbf{Y}_t - \mathbf{Y}_{ss}||/||\mathbf{Y}_0 - \mathbf{Y}_{ss}||$, where $||\cdot||$ is the Euclidean distance; $\mathbf{Y}_0$, $\mathbf{Y}_t$ and $\mathbf{Y}_{ss}$ are the initial, current and steady-state (final) concentration fields, respectively. The D-measures for the three time steps selected in Figure 5 are: 0.832 (20 s), 0.495 (60 s), 0.186 (100 s), representing different stages of the process dynamics. At all time steps, the GP-PCA prediction successfully emulates the major features of the concentration profile from simulation, though it slightly overestimates the concentration in some areas. GP-PCA is clearly more favourable than NN-PCA in this case study, for the prediction error of NN-PCA is significantly more appreciable.

(Figure 5 around here)

5. Concluding remarks

In view of the computational complexity of some process simulations, meta-modelling has been recognized as a viable solution for utilizing these complex simulations. This paper has extended the scope of meta-modelling for time-space-dependent output variables. The usual practice is to summarize the time-space-varying variables into a single or several outputs, such as operating profit when using flowsheet simulation for plant optimization, so that conventional meta-modelling becomes possible. However, when complex simulations are utilized beyond optimization tasks, direct meta-modelling of the high-dimensional time-space-dependent outputs is often desired. The proposed PCA-based approach, in conjunction with GP regression model, has been demonstrated to be effective and efficient for this purpose. Nevertheless, this study has been restricted to a small number of static input variables. When the inputs also become time-space-dependent, PCA may still be applicable for dimension reduction, provided that the inputs are highly correlated in the time-space coordinates. Otherwise, a large number of simulation runs may be unavoidable to collect sufficient computer data for meta-modelling. Further investigation on this topic should be conducted.

Currently we are focused on applying the meta-modelling methodology for the analysis of more realistic simulation of vapour cloud formation and explosion process, which is of high relevance to industrial and civilian safety assurance.

References

Bayarri, M.J., Berger, J.O., Cafeo, J., Garcia-Donato, G., Liu, F., Palomo, J., Parthasarathy, R.J., Paulo, R., Sacks, J., Walsh, D. (2007). Computer model validation with functional output, Annals of Statistics, 35, 1874-1906.

Brown, J. (2005). Inhalability at low wind speeds, Inhalation Toxicology, 17 (14), 831-837.

Caballero, J.A., Grossmann, I.E. (2008a). Rigorous flowsheet optimization using process simulators and surrogate models, in: Braunschweig, B. & Joulia, X. (eds.), Proc. 18th European Symposium on Computer Aided Process Engineering, pp. 551-556.

Caballero, J.A., Grossmann, I.E. (2008b). An algorithm for the use of surrogate models in modular flowsheet optimization, AIChE Journal, 54, 2633-2650.

Chen, T., Martin, E. (2009). Bayesian linear regression and variable selection for spectroscopic calibration, Analytica Chimica Acta, 631: 13-21.

Chen, T., Ren, J. (2009). Bagging for Gaussian process regression, Neurocomputing, 72, 1605-1610.
Chen, V.C.P., Tsui, K.L., Barton R.R., Meckesheimer, M. (2006). A review on design, modeling and applications of computer experiments, IIE Transactions, 38, 273-291.
Chryssolouris, G., Lee, M., Ramsey, A. (1996). Confidence interval prediction for neural network models, IEEE Transactions on Neural Networks, 7, 229-232.
Csato, L., Opper, M. (2002). Sparse on-line Gaussian processes, Neural Computation, 14, 641-668.
Dutournie, P., Salagnac, P., Glouannec, P. (2006). Optimization of radiant-convective drying of a porous medium by design of experiment methodology, Drying Technology, 24, 953-963.
Fang, K.T., Lin, D.K.J., Winker, P., Zhang, Y. (2000). Uniform design: theory and applications, Technometrics, 42, 237-248.
Gay, D., Ray, H. (1995). Identification and control of distributed parameter systems by means of the singular value decomposition, Chemical Engineering Science, 50, 1519-1539.
Gidaspow, D. (1994). Multiphase Flow and Fluidization: Continuum and Kinetic Theory Descriptions, Academic Press, London.
Gomes, M.V.C., Bogle, I.D.L., Biscaia Jr., E.C., Odloak, D. (2008). Using kriging models for real-time process optimisation, in: Braunschweig, B. & Joulia, X. (eds.), Proc. 18th European Symposium on Computer Aided Process Engineering, pp. 361-366.
Hangos, K., Cameron, I. (2001). Process Modelling and Model Analysis, Academic Press.
Holmes, C.C., Adams, N.M. (2002). A probabilistic nearest neighbour method for statistical pattern recognition, Journal of the Royal Statistical Society B (Statistical Methodology), 64, 295-306.
Hoo, K., Zhang, D. (2001). Low-order control-relevant models for a class of distributed parameter systems, Chemical Engineering Science, 56, 6683-6710.
Jolliffe, I. T. (2002). Principal Component Analysis, 2nd Edition. Springer.
Jones, D.R. (2001). A taxonomy of global optimization methods based on response surfaces, Journal of Global Optimization, 21, 345-383.
Kalagnanam, J.R., Diwekar, U.M. (1997). An efficient sampling technique for off-line quality control, Technometrics, 39, 308-319.
Kisa, M., Jelemensky, L. (2009). CFD dispersion modelling for emergency preparadnes, Journal of Loss Prevention in the Process Industries, 22, 97-104.
Kennedy, M.C., O'Hagan, A. (2001). Bayesian calibration of computer models, Journal of the Royal Statistical Society B, 63, 425-464.
Likar, B., Kocijan, J. (2007). Predictive control of a gas-liquid separation plant based on a Gaussian process model, Computers and Chemical Engineering, 31, 142-152.
McKay, M.D., Conover, W.J., Beckman, R.J. (1979). A comparison of three methods for selecting values of input variables in the analysis of output from a computer code, Technometrics, 21: 239-245.
MacKay, D.J.C. (1992). A practical Bayesian framework for back-propagation networks. Neural Computation, 4(3), 448-472.
Myers, R.H., Montgomery, D.C. (1995). Response Surface Methodology, Wiley.
O'Hagan, A. (2006). Bayesian analysis of computer code outputs: a tutorial, Reliability Engineering and Systems Safety, 91, 1290-1300.
Palmer, K., Realff, M. (2002). Metamodeling approach to optimization of steady-state flowsheet simulations: model generation, Chemical Engineering Research and Design, 80, 760-772.
Quiñonero-Candela, J., Rasmussen, C.E., Sinz, F., Bousquet, O., Schölkopf, B. (2006). Evaluating predictive uncertainty challenge, in: J. Quiñonero-Candela et al. (Eds.), Machine Learning Challenges, Lecture Notes in Computer Science, vol. 3944, Springer, pp. 1-27.
Rasmussen, C.E., Williams, C.K.I. (2006). Gaussian Processes for Machine Learning, MIT Press.
Romijn, R., Ozkan, L., Weiland, S., Ludlage, J., Marquardt, W. (2008). A grey-box modelling approach for the reduction of nonlinear systems, Journal of Process Control, 18, 906-914.
Rougier, J. (2008). Efficient emulators for multivariate deterministic functions, Journal of Computational and Graphical Statistics, 17, 827-843.
Sacks, J., Welch, W.J., Mitchell, T.J., Wynn, H.P. (1989). Design and analysis of computer experiments, Statistical Science, 4, 409-435.
Shvartsman, S., Theodoropoulos, C., Rico-Martinez, R., Kevrekidis, I., Titi, E., Mountziaris, T. (2000).

Order reduction for nonlinear dynamic models of distributed reacting systems, Journal of Process Control, 10, 177-184.

Tang, Q., Lau, Y.B., Hu, S., Yan, W., Yang, Y., Chen T. (2010). Response surface methodology using Gaussian processes: towards optimizing the *trans*-stilbene epoxidation over Co^{2+}-NaX catalysts, Chemical Engineering Journal, 156, 423-431.

Tipping, M.E., Bishop, C.M. (1999). Probabilistic principal component analysis, Journal of the Royal Statistical Society B, 61, 611-622.

Thodberg, H.H. (1996). Review of Bayesian neural networks with an application to near infrared spectroscopy, IEEE Transactions on Neural Networks, 7(1), 56-72.

Venkatasubramanian, V., Rengaswamy, R., Kavuri, S.N., Yin, K. (2003). A review of process fault detection and diagnosis Part III: Process history based methods, Computers and Chemical Engineering, 27, 327-346.

Yuan, J., Wang, K., Yu, T., Fang, M. (2008). Reliable multi-objective optimization of high-speed WEDM process based on Gaussian process regression, International Journal of Machine Tools and Manufacture, 48, 47-60.

Zhang, J., Morris, A.J., Martin, E.B., Kiparissides, C. (1998). Prediction of polymer quality in batch polymerisation reactors using robust neural networks, Chemical Engineering Journal, 69, 135-143.

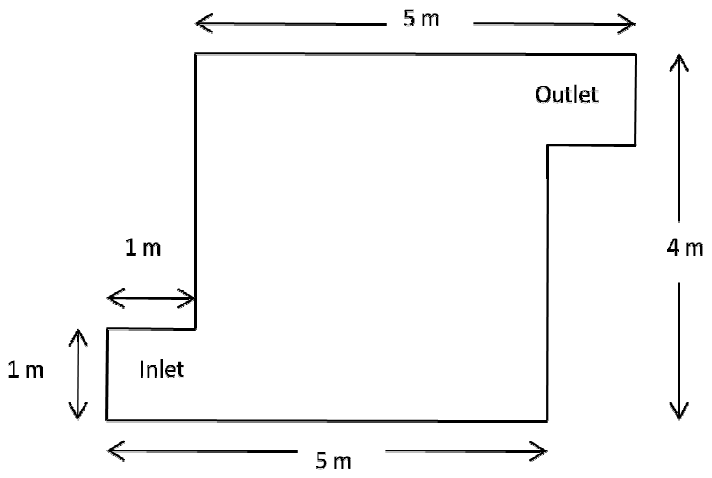

Figure 1. The geometry used for the CFD simulation.

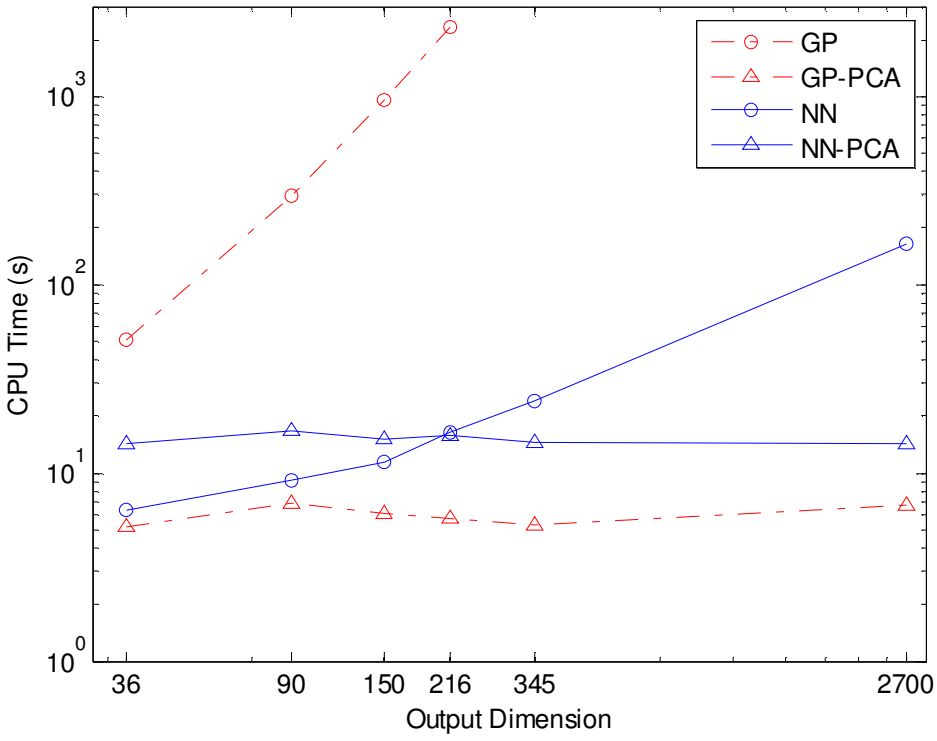

Figure 2. Computation time versus output dimension. The CPU time is based on developing one meta-model using 19 runs of simulation data as in the LOOCV procedure.

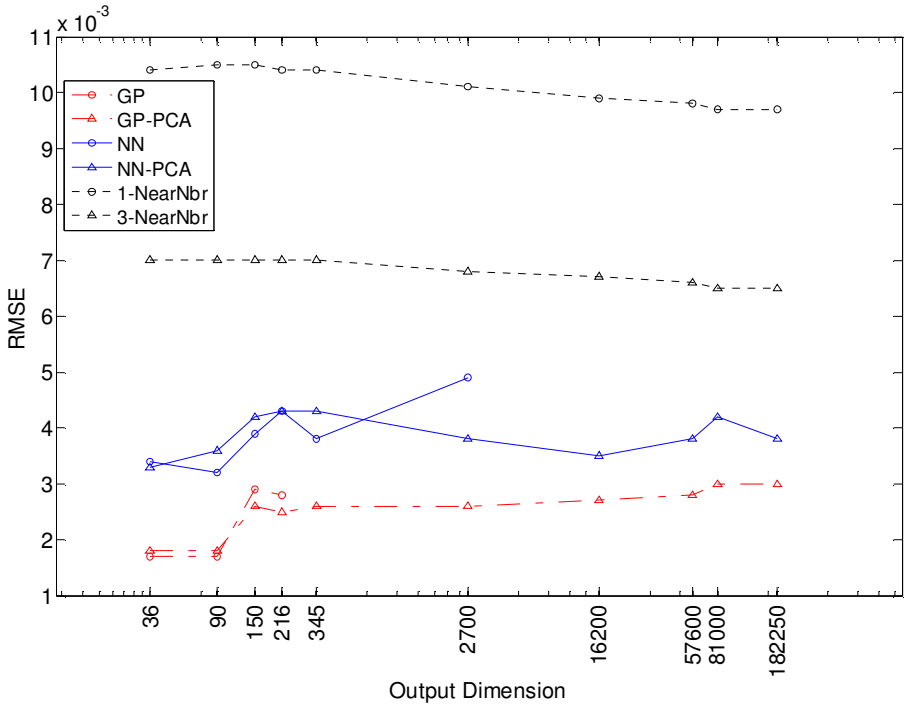

Figure 3. RMSE of cross-validation versus output dimension.

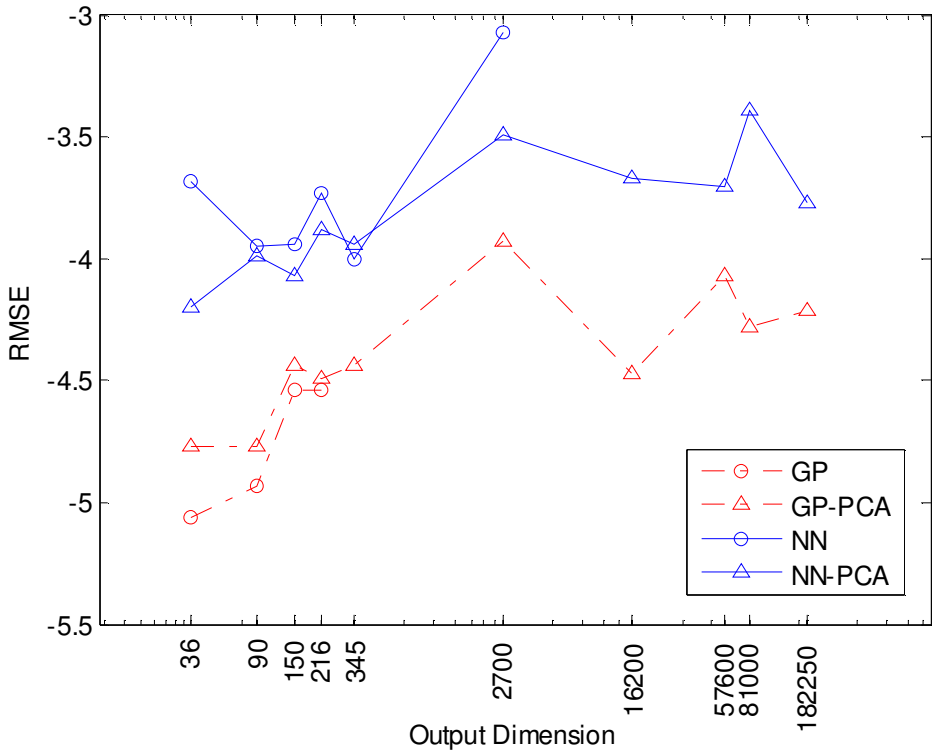

Figure 4. NLPD of cross-validation versus output dimension.

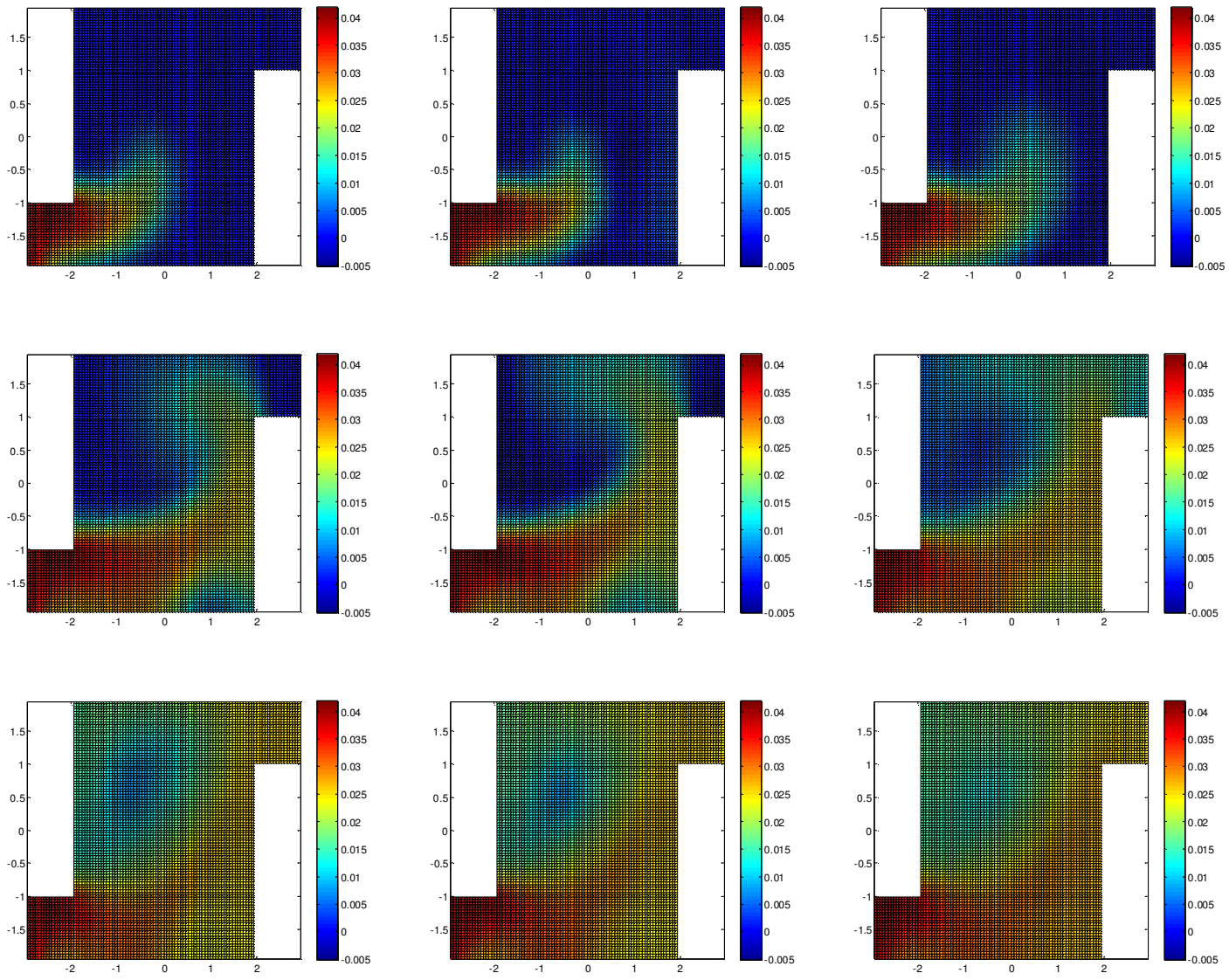

Figure 5. Prediction of one simulation run from Case No. 7 in Table 1. Rows correspond to time instances: 20 s (upper), 60 s (middle) and 100 s (lower). Columns correspond to different methods: Simulation (left), GP-PCA (middle) and NN-PCA (right).

Table 1: The settings of output dimension under investigation.

Case No.	1	2	3	4	5	6	7	8	9	10
Time	2	2	3	3	3	6	9	18	45	45
Space	18	45	50	72	115	450	1800	3200	1800	4050
Total	36	90	150	216	345	2700	16200	57600	81000	182250

Table 2: RMSE for the prediction of the first eight principal components (Case No. 7 in Table 1).

Component	1	2	3	4	5	6	7	8
RMSE	0.099	0.260	0.396	0.994	0.928	2.182	2.022	4.426