entropy

MDPI

Article

Sequential Batch Design for Gaussian Processes Employing Marginalization †

Roland Preuss * and Udo von Toussaint

Max-Planck-Institute for Plasma Physics, 85748 Garching, Germany; udt@ipp.mpg.de

* Correspondence: preuss@ipp.mpg.de

† This paper is an extended version of our paper published in 36th International Workshop on Bayesian Inference and Maximum Entropy Methods in Science and Engineering, Ghent, Belgium, 10–15 July 2016.

Academic Editor: Geert Verdoolaege

Received: 30 November 2016; Accepted: 17 Feburary 2017; Published: 21 Feburary 2017

Abstract: Within the Bayesian framework, we utilize Gaussian processes for parametric studies of long running computer codes. Since the simulations are expensive, it is necessary to exploit the computational budget in the best possible manner. Employing the sum over variances —being indicators for the quality of the fit—as the utility function, we establish an optimized and automated sequential parameter selection procedure. However, it is also often desirable to utilize the parallel running capabilities of present computer technology and abandon the sequential parameter selection for a faster overall turn-around time (wall-clock time). This paper proposes to achieve this by marginalizing over the expected outcomes at optimized test points in order to set up a pool of starting values for batch execution. For a one-dimensional test case, the numerical results are validated with the analytical solution. Eventually, a systematic convergence study demonstrates the advantage of the optimized approach over randomly chosen parameter settings.

Keywords: parametric studies; Gaussian process; parallelization; batch execution

PACS: 02.50.-r; 52.65.-y

1. Introduction

The generation of optimal simulation input points is of particular interest for long running computer codes. As in the field of plasma–wall interactions of fusion plasmas, the running time of the simulation codes is in the order of months, so the input parameter settings should better be chosen well. Though considerable parallelization efforts have been spent to accelerate the code, the speed-up is limited to a certain margin, from which point on it is futile to utilize more and more processor cores. Apart from this drawback, one would still have to wait for a sequential execution of one parameter setting after the other, since the next best set of input parameters depends on the previous result. This paper overcomes this by proposing several most promising parameter setups which are obtained by marginalizing over optimal input points. The pool of such independent starting sets enables the parallel execution of the original simulation code.

The problem of predicting function values in a multi-dimensional space supported by given data is a regression problem for a non-trivial function of unknown shape. Given n input data vectors $\boldsymbol{x}_i$ of dimension N_{dim} (with matrix $\boldsymbol{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_n)$) and corresponding target data $\boldsymbol{y} = (y_1, ..., y_n)^T$ blurred by Gaussian noise of variance σ_d^2, the sought quantity is the target value f_* at test input vector $\boldsymbol{x}_*$. The latter would be generated by a function $f(\boldsymbol{x})$

$$y = f(\boldsymbol{x}) + \epsilon \quad , \tag{1}$$

where $\langle \epsilon \rangle = 0$ and $\langle \epsilon^2 \rangle = \sigma_d^2$. Since we are completely ignorant about a model describing function, our ansatz is to employ the Gaussian process method, with which any uniformly continuous function may be represented. As a statistical process, it is fully defined by its covariance function and called Gaussian, because any collection of random variables produced by this process has a Gaussian distribution.

The Gaussian process method defines a distribution over functions. One can think of the analysis as taking place in a space of functions (function-space view), which is conceptually different from the familiar view of solving the regression problem of, for instance, the standard linear model (SLM)

$$f^{\mathrm{SLM}}(\boldsymbol{x}) = \boldsymbol{x}^T \boldsymbol{w} \quad , \tag{2}$$

in the space of the weights $\boldsymbol{w}$ (weight-space view). At this point, it is instructive to restate the results for the latter: the predictive distribution depending on mean $\bar{f}_*$ and variance for a test input data point $\boldsymbol{x}_*$ is given by

$$p(f_*^{\mathrm{SLM}} | \boldsymbol{X}, \boldsymbol{y}, \boldsymbol{x}_*) \propto \mathcal{N}\left(\bar{f}_*^{\mathrm{SLM}}, \mathrm{var}(f_*^{\mathrm{SLM}})\right) \quad , \tag{3}$$

with

$$\bar{f}_*^{\mathrm{SLM}} = \frac{1}{\sigma_d^2} \boldsymbol{x}_*^T \left[\sigma_d^{-2} \boldsymbol{X}\boldsymbol{X}^T + \Sigma_p^{-1}\right]^{-1} \boldsymbol{X}\boldsymbol{y} \quad , \tag{4}$$

$$\mathrm{var}(f_*^{\mathrm{SLM}}) = \boldsymbol{x}_*^T \left[\sigma_d^{-2} \boldsymbol{X}\boldsymbol{X}^T + \Sigma_p^{-1}\right]^{-1} \boldsymbol{x}_* \quad . \tag{5}$$

Σ_p is the covariance in a Gaussian prior on the weights. In the next section, these results will be transferred to the function-space view of the Gaussian process method.

The Gaussian process method has been greatly appreciated in the fields of neural networks and machine learning [1–5]. Residing on this, further work showed the applicability of active data selection via variance-based criteria [6,7]. In general, for unknown functions that are costly to evaluate, Bayesian optimization [8] was deployed with either sequential [9,10] or batch design [11], and recently in combination of both [12,13]. All these approaches draw the next-best data points from a surrogate model function provided by Gaussian processes in order to optimize the description of this unknown function. Our paper addresses a different question, since we ask for the next best parameter set after next for which this costly function should be evaluated; i.e., without having the knowledge of the result for the next best point already at hand. Additionally, it differs fundamentally in the handling of results of the surrogate model. This papers proposes a marginalization approach which integrates over all possible surrogate model values at the next-best point to measure at, while former approaches add the Gaussian process data to the data pool, constituting the basis for the succeeding analysis.

The paper is organized at follows. Sections 2 and 3 restate the results of the analysis in Chapters 2.2, 2.3, and 5.4 of the book of Rasmussen and Williams [14], the notation of which we follow, apart from small amendments. Section 4 together with Appendix B show an autonomous optimization algorithm for sequential selection of further parameter sets, as known from literature (e.g., [9]). In order to accomplish an efficient batch design, we propose a marginalization approach in Section 5, which is validated for a one-dimensional example in Section 6. Section 7 is independent from the marginalization approach, and studies the convergence behaviour of the optimization algorithm of Section 4. A compilation of the variable names may be found in Appendix A.

2. Prediction of Function Values

As stated above, the defining quantity of the Gaussian process method is the covariance function. Its choice is decisive for the inference we want to apply. It is where we incorporate all the properties that we would like our (hidden) function describing our problem to have in order to influence the result. For example, the neighbourhood of two input data vectors $\boldsymbol{x}_p$ and $\boldsymbol{x}_q$ should be of relevance for the smoothness of the result. This shall be expressed by a length scale λ which represents the long-range

dependence of the two vectors. For the covariance function itself, we employ a Gaussian-type exponent with the negative squared value of the distance between two vectors $\boldsymbol{x}_p$ and $\boldsymbol{x}_q$

$$k(\boldsymbol{x}_p, \boldsymbol{x}_q) = \sigma_f^2 \exp\left\{-\frac{1}{2}\left|\frac{\boldsymbol{x}_p - \boldsymbol{x}_q}{\lambda}\right|^2\right\}. \tag{6}$$

σ_f^2 is the signal variance. If one is ignorant about this value, literature proposes to set it to one as default value (Chapters 2.3 and 5.4 in [14]). However, in probability theory, we consider it as a hyper-parameter to be marginalized over (see next chapter). To avoid lengthy formulae, we abbreviate the covariance matrix of the input data as $(\boldsymbol{K})_{ij} = k(\boldsymbol{x}_i, \boldsymbol{x}_j)$ and the vector of covariances between test point and input data as $(\boldsymbol{k}_*)_i = k(\boldsymbol{x}_*, \boldsymbol{x}_i)$.

Moreover, we consider the degree of information which the data contain by a term $\sigma_n^2\boldsymbol{\Delta}$ to be composed of an overall variance σ_n^2 accounting that the data are noisy and the matrix $\boldsymbol{\Delta}$ with the variances σ_d^2 of the given input data on its diagonal, and zero otherwise. While σ_n^2 is a hyper-parameter, the matrix entry $(\sigma_d)_i$ is the relative uncertainty estimation of a single data point y_i and is provided by the experimentalist. If no uncertainties of the input data are given, $\boldsymbol{\Delta}$ is set to the identity matrix. It can be shown (Chapter 2.2 in [14]) that in analogy to Equation (3) for given λ, σ_f and σ_n the probability distribution for a single function value f_* at test input $\boldsymbol{x}_*$ is

$$p(f_*|\boldsymbol{X}, \boldsymbol{y}, \boldsymbol{x}_*) \propto \mathcal{N}\left(\bar{f}_*, \mathrm{var}(f_*)\right) \quad , \tag{7}$$

with mean

$$\bar{f}_* = \boldsymbol{k}_*^T \left(\boldsymbol{K} + \sigma_n^2\boldsymbol{\Delta}\right)^{-1} \boldsymbol{y} \tag{8}$$

and variance

$$\mathrm{var}(f_*) = \boldsymbol{k}(\boldsymbol{x}_*, \boldsymbol{x}_*) - \boldsymbol{k}_*^T \left(\boldsymbol{K} + \sigma_n^2\boldsymbol{\Delta}\right)^{-1} \boldsymbol{k}_* \quad . \tag{9}$$

3. Marginalizing the Hyper-Parameters

The hyper-parameters $\boldsymbol{\theta} = (\lambda, \sigma_f, \sigma_n)^T$ determine the result of the Gaussian process method. Since we do not know a priori which setting is useful, we marginalize over them later on in order to get the target values $\boldsymbol{f}_*$ for test inputs $\boldsymbol{X}_*$. Their moments are

$$\langle \boldsymbol{\theta}^m \rangle = \frac{1}{Z}\int \mathrm{d}\boldsymbol{\theta}\, \boldsymbol{\theta}^m p(\boldsymbol{\theta}|\boldsymbol{y}) = \frac{1}{Z}\int \mathrm{d}\boldsymbol{\theta}\, \boldsymbol{\theta}^m p(\boldsymbol{y}|\boldsymbol{\theta}) p(\boldsymbol{\theta}) \quad , \qquad Z = \int \mathrm{d}\boldsymbol{\theta}\, p(\boldsymbol{\theta}|\boldsymbol{y}), \tag{10}$$

where our special interest is the first (expectation value) and second central (variance or rather square root thereof; i.e., standard deviation) moment listed in all subsequent tables.

For the choice of the prior, not much is to be expected. A sensible choice would be to assume them in the order of one with a variance of the same size, but confined to be positive

$$p(\theta_i) \propto \mathcal{N}(1,1) \;\; \forall \;\; \theta_i \geq 0\,, \quad \text{and} \quad p(\theta_i) = 0 \;\; \text{otherwise.} \tag{11}$$

Depending on the application, one should check on these assumptions and be cautious that the prior of the hyper-parameters should not influence the result.

The marginal likelihood $p(\boldsymbol{y}|\boldsymbol{\theta})$ is obtained by

$$p(\boldsymbol{y}|\boldsymbol{\theta}) = \int \mathrm{d}\boldsymbol{f}\; p(\boldsymbol{y}|\boldsymbol{f}, \boldsymbol{\theta}) p(\boldsymbol{f}|\boldsymbol{\theta}). \tag{12}$$

As we deal with the Gaussian process, the probability functions are of Gaussian type, with the likelihood as $p(\boldsymbol{y}|\boldsymbol{f},\boldsymbol{\theta}) \propto \mathcal{N}(\boldsymbol{f},\sigma_n\boldsymbol{\Delta})$ and the prior for $\boldsymbol{f}$ as $p(\boldsymbol{f}|\boldsymbol{\theta}) \propto \mathcal{N}(\boldsymbol{0},\boldsymbol{K})$ (end of Chapter 2.2 in [14]). Thus, the integration in Equation (12) yields

$$\log p(\boldsymbol{y}|\boldsymbol{\theta}) = \text{const} - \frac{1}{2}\boldsymbol{y}^T\left[\boldsymbol{K}(\boldsymbol{\theta}) + \sigma_n^2\boldsymbol{\Delta}\right]^{-1}\boldsymbol{y} - \frac{1}{2}\log\left|\boldsymbol{K}(\boldsymbol{\theta}) + \sigma_n^2\boldsymbol{\Delta}\right|. \tag{13}$$

The expectation value for the targets $\boldsymbol{f}_*$ at test inputs $\boldsymbol{X}_*$ employs the marginal likelihood and priors for the hyper-parameters from above

$$\langle \boldsymbol{f}_* \rangle = \int \mathrm{d}\boldsymbol{\theta}\; \bar{\boldsymbol{f}}_* \frac{p(\boldsymbol{y}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{\int \mathrm{d}\boldsymbol{\theta}'\; p(\boldsymbol{y}|\boldsymbol{\theta}')p(\boldsymbol{\theta}')} \quad , \tag{14}$$

where the fraction contains the sampling density in Markov chain Monte Carlo.

Rescaling of the input data and whitening of the output is performed in order to do the analysis unhampered by large scales or biased from a linear trend. All data has been back-transformed for display.

4. Closed Loop Optimization Scheme

With the help of the formulas in Equations (8) and (9), we can ask for an arbitrary target value and its variance within some region of interest (ROI) substantiated by existing data. In order to determine the next-best points at which to perform an expensive experiment or to run a long-term computer code, we propose the following autonomous optimization algorithm for sequential parameter selection [15].

Since the variance in Equation (7) depends only on the input $\boldsymbol{X}$ and not on the target data, we can immediately evaluate the utility of a further datum without the need to marginalize over the (unknown) target outcome. To achieve this, one has to iterate within a region of interest $\mathcal{I}$ set by the experimentalist over each grid point $\boldsymbol{\xi} \in \mathcal{I}$, which is tentatively handled as being part of the pool of input data vectors. Then, for all test inputs $\boldsymbol{x}_* \in \mathcal{I}$, the resulting variances var′ have to be determined according to Equation (7), but with changed $(N+1)\times(N+1)$ covariance matrix $\boldsymbol{K}'$ of the expanded input $\boldsymbol{X}' = \{\boldsymbol{X},\boldsymbol{\xi}\}$. Summing up the variances var′ over all grid points in $\mathcal{I}$ provides a measure for the utility $U(\boldsymbol{\xi})$ of a target data obtained at input vector $\boldsymbol{\xi}$:

$$U(\boldsymbol{\xi}) = -\sum_{\boldsymbol{x}_*\in\mathcal{I}} \mathrm{var}'(f_*). \tag{15}$$

The minus sign in Equation (15) reflects the fact that a smaller sum over the variances is connected with a higher utility of the additional datum. The goal is to find $\boldsymbol{\xi}_{\max}$ with the largest utility:

$$\boldsymbol{\xi}_{\max} = \arg\max_{\{\boldsymbol{\xi}\}} U(\boldsymbol{\xi}). \tag{16}$$

At the obtained $\boldsymbol{\xi}_{\max}$, the next measurement is most informative. If the target data is produced by a computer code, one can set up an autonomously running procedure which invokes the computer code at $\boldsymbol{\xi}_{\max}$ to add the next target outcome to the data pool, with which the search for the next most informative point is performed. This scheme would be repeated in an iterative manner, until the increase in information from an additional target datum drops below some predefined level or becomes insignificant.

5. Marginalizing Test Points

The above iterative algorithm proposes further input settings for future evaluations by an automated but sequential procedure. For applications with long running computer codes, it would be desirable to exploit the parallel capabilities of modern multi-processor systems in order to accelerate this process. We propose the creation of different onsets for batch operation by invoking a

marginalization procedure that simply marginalizes successively over as many test points as batch processors are available. The first processor would be fed the original problem with the already obtained $N_{\text{start}} = N$ input points. For the second processor, we marginalize over the target value f_1 at a first test input vector $\hat{\boldsymbol{x}}_1$ chosen by the estimation criterion of the previous section; i.e., which test point lowers the variance of the full system the most according to Equations (15) and (16). The optimal input vector $\boldsymbol{x}_{\text{opt}} = \boldsymbol{\xi}_{\max}$ found is then identified with the input vector $\hat{\boldsymbol{x}}_1$, for which the target value has to marginalized. Then, the probability distribution of the next test point f_{1*} is given by the marginalization rule

$$\begin{aligned} p(f_{1*}|\boldsymbol{X},\boldsymbol{y},\boldsymbol{x}_{1*}) &= \int \mathrm{d}f_1 p(f_{1*},f_1|\boldsymbol{X},\boldsymbol{y},\boldsymbol{x}_{1*},\hat{\boldsymbol{x}}_1) \\ &= \int \mathrm{d}f_1 p(f_{1*}|\boldsymbol{X},\boldsymbol{y},\boldsymbol{x}_{1*},\hat{\boldsymbol{x}}_1,f_1) p(f_1|\boldsymbol{X},\boldsymbol{y},\hat{\boldsymbol{x}}_1), \end{aligned} \tag{17}$$

where

$$p(f_{1*}|\boldsymbol{X},\boldsymbol{y},\boldsymbol{x}_{1*},\hat{\boldsymbol{x}}_1,f_1) \propto \mathcal{N}\left(\bar{f}_{1*}, \mathrm{var}(f_{1*})|_{f_1}\right), \tag{18}$$

$$p(f_1|\boldsymbol{X},\boldsymbol{y},\hat{\boldsymbol{x}}_1) \propto \mathcal{N}\left(\bar{f}_1, \mathrm{var}(f_1)\right). \tag{19}$$

We now rewrite Equation (8) by $\bar{f}_{1*} = \boldsymbol{\kappa}_{1*}^T \boldsymbol{y}_1$, with $\boldsymbol{\kappa}_{1*}^T = \boldsymbol{k}_1^T \left(\boldsymbol{K} + \sigma_n^2 \boldsymbol{\Delta}\right)^{-1}$ and $\boldsymbol{y}_1^T = (\boldsymbol{y}^T, f_1)$. For the calculation of Equation (17), it is expedient to separate the last entry in the κ-vector: $\boldsymbol{\kappa}_{1*}^T = (\hat{\boldsymbol{\kappa}}_{1*}^T, (\boldsymbol{\kappa}_{1*})_{N+1})$. The marginalization integral only has to handle Gaussians, and readily results in the predictive distribution for the Gaussian process regression for one marginalized test point f_1

$$p(f_{1*}|\boldsymbol{X},\boldsymbol{y},\boldsymbol{x}_{1*}) \propto \mathcal{N}\left(\boldsymbol{\kappa}_{1*}^T \bar{\boldsymbol{y}}_1, \mathrm{var}(f_{1*})|_{f_1}\right), \tag{20}$$

with $\bar{\boldsymbol{y}}_1^T = (\boldsymbol{y}^T, \bar{f}_1)$ and the marginalized variance

$$\mathrm{var}(f_{1*})|_{f_1} = \mathrm{var}(f_{1*}) + (\boldsymbol{\kappa}_{1*})_{N+1}^2 \mathrm{var}(f_1). \tag{21}$$

While the vector multiplication for the mean in Equation (20) is just the result without marginalization enlarged by the expectation of the marginalized value, the variance has acquired an additional term in Equation (21) compared to the case without marginalization $\mathrm{var}(\bar{f}_{1*})$. This is reasonable because the target data pool has increased by an expectation value $\bar{f}_1$ based on the very same data pool, so the lack of new information can only result in a larger variance.

The whole marginalization procedure is easily extended to obtain the successive points f_2 to $f_{N_{\text{marg}}}$ to be marginalized over. As we intend to use the marginalization procedure to create a pool of parameter setups to start a long running simulation code in parallel—but for the "most valuable" setups only—the order of the successively created setups is of importance.

$$\begin{aligned} \mathrm{var}(f_{2*})|_{f_1,f_2} &= \mathrm{var}(f_{2*}) + (\boldsymbol{\kappa}_{2*})_{N+1}^2 \mathrm{var}(f_2)|_{f_1} \quad , \\ &\vdots \\ \mathrm{var}(f_{N_{\text{marg}}*})|_{f_1,\ldots,f_{N_{\text{marg}}}} &= \mathrm{var}(f_{N_{\text{marg}}*}) + (\boldsymbol{\kappa}_{N_{\text{marg}}*})_{N+1}^2 \mathrm{var}(f_{(N_{\text{marg}}-1)*})|_{f_1,\ldots,f_{N_{\text{marg}}-1}} \quad . \end{aligned} \tag{22}$$

Once the variances are calculated, the largest utility can be obtained by using Equations (15) and (16).

6. Validation in One Dimension

In order to examine the dependence of the result on the marginalized points, we simulate a one-dimensional test case that is analytically accessible.

For an input data set consisting of a single point at x_1 with target value y_1, the expectation value at test point x_* is

$$\bar{f}_* = \frac{\sigma_f^2}{\sigma_f^2 + \sigma_n^2 \sigma_{d_1}^2} \exp\left\{ -\frac{1}{2} \left| \frac{x_1 - x_*}{\lambda} \right|^2 \right\} y_1 \tag{23}$$

with variance

$$\mathrm{var}(f_*) = \sigma_f^2 - \frac{\sigma_f^2}{\sigma_f^2 + \sigma_n^2 \sigma_{d_1}^2} \exp\left\{ - \left| \frac{x_1 - x_*}{\lambda} \right|^2 \right\} y_1. \tag{24}$$

This is plotted in Figure 1a with $x_1 = 0$. The y-value is taken from the sinusoidal model shown (a cos-function with decreasing amplitude), however subjected to whitening, which results in this simple case of one point to the assignment of a zero value. The back-transformation to the scale of the input target value leads to the horizontal line titled "Prediction" (marked by small white circles). As can be seen from Equation (24), the variance increases with the distance from the data point. The utility according to Equation (15) was scaled and normalized to fit within the range $[-3, -2]$ of each graph. It produces the result of $x_{\mathrm{opt}} = 0.775$ for the next best point to measure at. Without marginalization, this would be the input datum returning the results of Figure 1b for our simple one-dimensional model. After ten iterations, the prediction is—within the gray uncertainty region—almost pinned down to the model generating function (see Figure 1c).

To run the whole process on a second processor, one would take the first optimal point at $\hat{x}_1 = 0.775$ for the marginalization step; i.e., its target value f_1 does not enter the data base. The prediction can also be done analytically to give

$$\bar{f}_{1*} = \frac{\sigma_f^2}{\sigma_f^2 + \sigma_n^2 \sigma_{d_1}^2} \exp\left\{ -\frac{1}{2} \left| \frac{x_1 - x_{1*}}{\lambda} \right|^2 \right\} y_1 + \frac{\sigma_f^2}{\sigma_f^2 + \sigma_n^2 \mathrm{var}(\bar{f}_*)} \exp\left\{ -\frac{1}{2} \left| \frac{x_* - x_{1*}}{\lambda} \right|^2 \right\} \bar{f}_*, \tag{25}$$

with variance

$$\mathrm{var}(f_{1*}) = \sigma_f^2 - \frac{\left(\sigma_f^2 + \sigma_n^2 \mathrm{var}(\bar{f}_*)\right) \mathrm{e}^{-\left|\frac{x_1 - x_{1*}}{\lambda}\right|^2} - 2\sigma_f^2 \mathrm{e}^{-\frac{(x_* - x_{1*})^2 + (x_1 - x_*)^2 + (x_1 - x_{1*})^2}{2\lambda^2}} + \left(\sigma_f^2 + \sigma_n^2 \sigma_{d_1}^2\right) \mathrm{e}^{-\left|\frac{x_* - x_{1*}}{\lambda}\right|^2}}{\left(\sigma_f^2 + \sigma_n^2 \sigma_{d_1}^2\right) \left(\sigma_f^2 + \sigma_n^2 \mathrm{var}(\bar{f}_*)\right) / \sigma_f^4 - \mathrm{e}^{-\left|\frac{x_* - x_1}{\lambda}\right|^2}}. \tag{26}$$

As can be seen in Figure 1d, the marginalized target point does not affect the prediction much (both y_1 and $\bar{f}_*$ are zero after whitening, and yield a zero line according to Equation (25)). Moreover, the standard deviation of the marginalized point, $\sqrt{\mathrm{var}(f_{1*})} = 0.3895$, is nearly four times higher than the inputted measurement uncertainty of $\sigma_{d_1} = 0.1$, which emphasizes the fact that the marginalized point is of less importance. The utility in Figure 1d gives rise to the next best point for a measurement at $x_{\mathrm{opt},1} = -0.8$, but after obtaining the target value, the next utility in Figure 1e already has its maximum at $x_{\mathrm{opt},2} = 0.7$, close to the position of the marginalized point $x_{\mathrm{marg}} = 0.775$. Together with further points obtained by the optimization procedure, this eventually concludes to a close similarity of the predicted curve with the model (see Figure 1f). The marginalized point is of no importance, as should be the case.

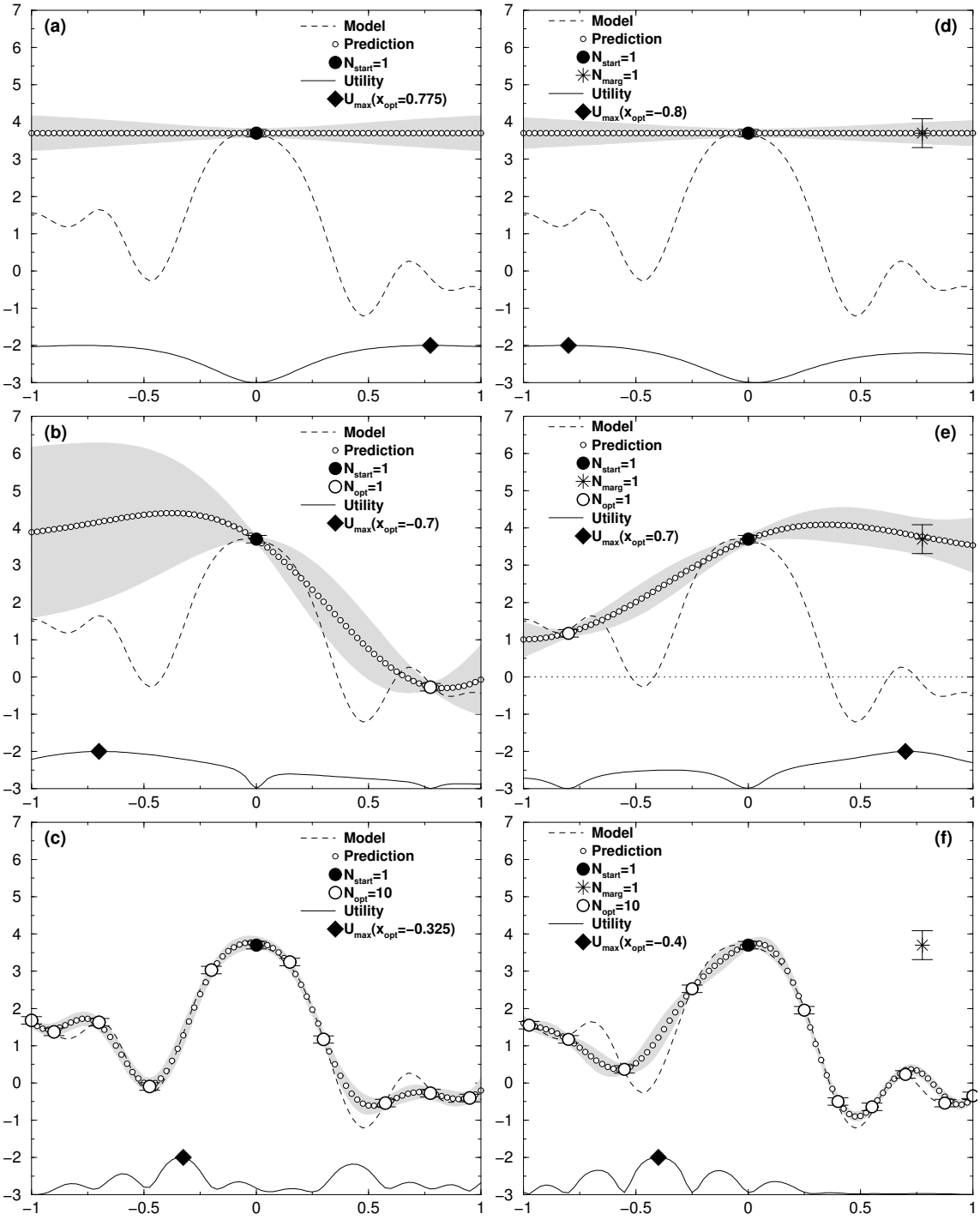

Figure 1. One-dimensional test case: Expectation values of the target function (Prediction) from Markov chain Monte Carlo (MCMC)-calculation where the grey-shaded area represents the uncertainty range. The utility (scaled and normalized) is plotted at the bottom of each figure. Its maximum $U_{\max}(x_{\text{opt}})$ shows the next input vector added to the pool of data. Top to bottom: increasing number of optimized points in input data pool. (**a–c**) without marginalized point. (**d–f**) with marginalized point at $\hat{x}_1 = 0.775$.

7. Convergence Study

In this final section, we investigate the convergence behaviour of the results produced by the employed Gaussian process method as a function of an increasing data pool. The quantity to be

studied is the total difference between the target values f_* at all test input vectors $\boldsymbol{\xi}$ in the region of interest $\mathcal{I}$ and the model outcome for these points

$$\text{deviation} = \frac{1}{N_{\text{ROI}}} \sum_{N_{\text{ROI}}} |f_* - f(\boldsymbol{\xi})|. \tag{27}$$

To begin with, we show that the marginalization procedure does not affect the result based on the increasing input data pool. This is supported by Figure 2, depicting the deviation of the prediction from the model as a function of the number of optimized points in the input data. Both batch runs—with and without marginalization—show the same quadratic descent.

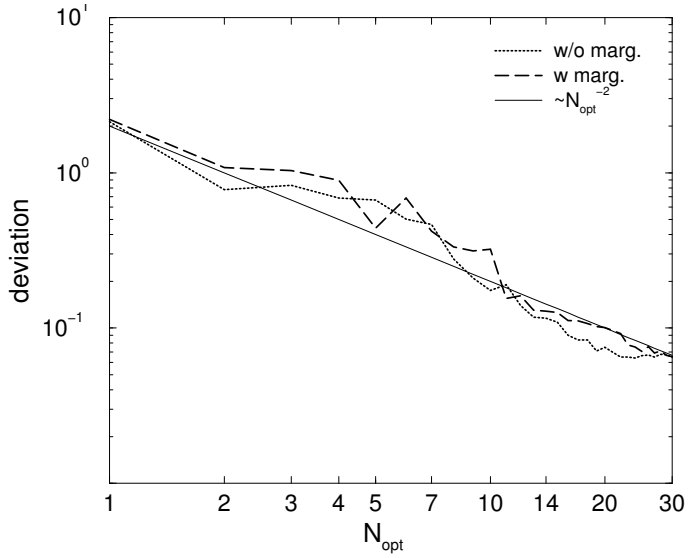

Figure 2. Deviation of the Gaussian process results from the exact model outcome as function of the number N_{opt} of obtained data at optimized parameter settings. Dotted line: without marginalized input values; dashed line: with one marginalized input value. Both show quadratic decay behaviour.

To demonstrate the advantage of the optimized approach over randomly chosen parameter settings, we run two test cases in one and two dimensions (see Figures 3 and 4) with different choices of the data uncertainties ranging from 10% over 1% to 0.1% of the total amplitude. Test case one consists of a simple Gaussian with a broad structure, test case two of a damped cosine function sitting on an inclined plane. To avoid artefacts originating from symmetry, the model function was shifted by a value of 0.2 from the center of the ROI. Of interest is the reduction of the variance with the number of acquired data points.

Let us first examine the one-dimensional case shown in Figure 3. The top row shows the two model functions (thin line), and the result of the Gaussian process method obtained after (a) $N_{\text{opt}} = 5$ and (d) $N_{\text{opt}} = 15$ data points were added to the data pool either as the result of the optimization procedure stated in Section 4 (thick line) or just by randomly chosen parameter settings (dashed line). The result from the optimized approach already covers all features of the respective model function, and is in good agreement with it. The random approach apparently did not choose an input vector close to one in Figure 3d, and misses the damped progress to the right.

The four Figure 3b,c,e,f show the convergence behaviour of the deviation Equation (27). Since the hyper-parameters λ, σ_f, and σ_n determine the outcome of the Gaussian process method (see Section 3) and vary a lot in the starting phase of the data acquisition, one has to wait until they have settled. They are plotted in the top of each graph. We restrain from showing σ_f, because it hardly varies for the test cases used. One finds that the hyper-parameters stabilize at about 30 data values added for the damped cosine model, and about 20 for the broad Gaussian. From thereon, the decay of the deviation

with the increasing data pool clearly shows square root behavior, while up to this settlement, higher power laws are in charge. The actual values of these higher powers for the decay seem to be somewhat arbitrary, but they are still substantial over all test cases to claim that the optimized approach for finding the best next parameter settings is much more fruitful than to set it up randomly.

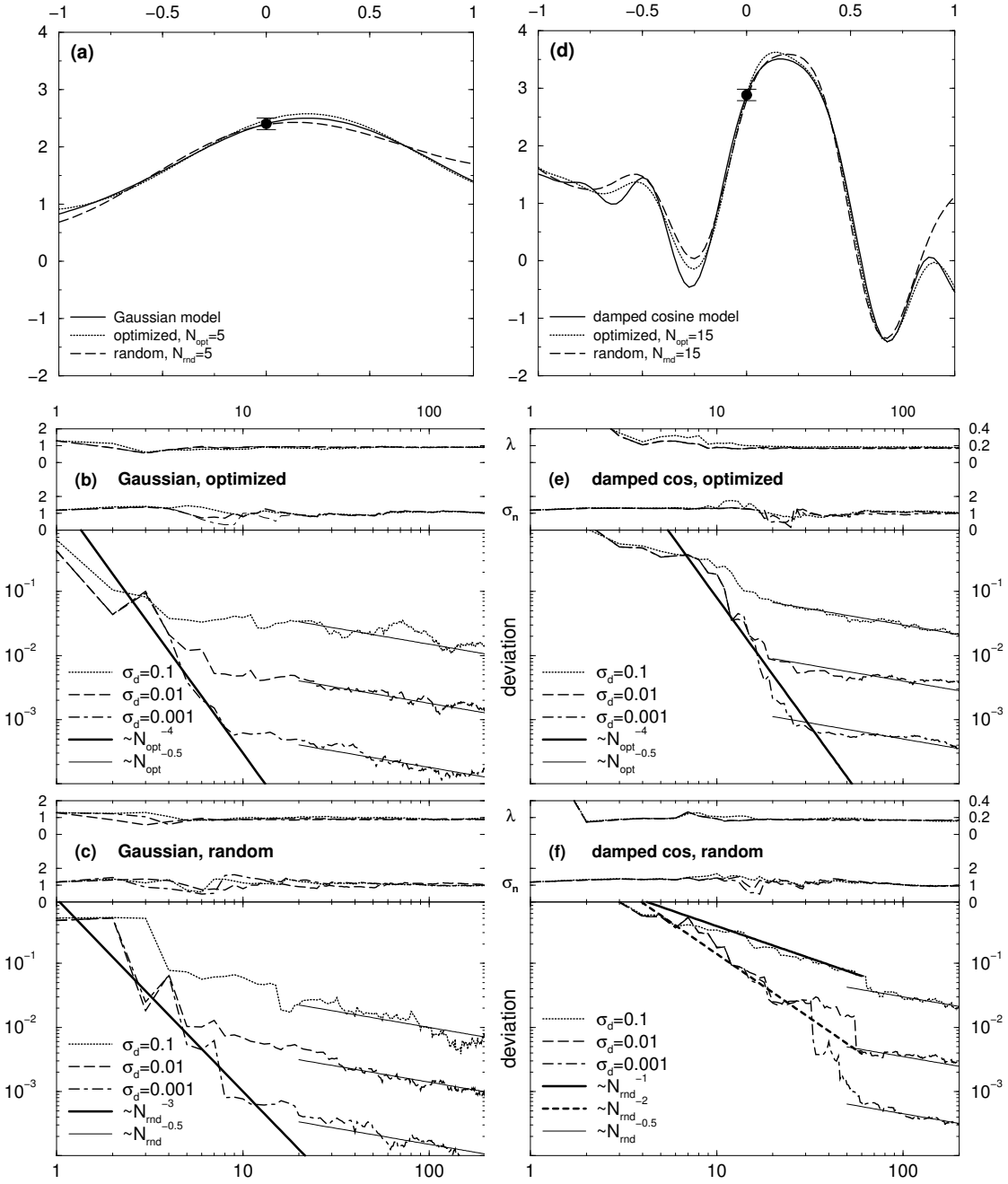

Figure 3. Two test cases for one-dimensional input vectors, N_{ROI}=81. (**a**–**c**) Gaussian model; (**d**–**f**) damped cosine model. Top row (**a**,**d**): model (solid line), initial input data value (filled circle), optimized approach (dotted line) vs. randomly chosen parameter settings (dashed line). Panels (**b**,**c**,**e**,**f**) with number of added points to the right: in the top of each figure, hyper-parameters λ and σ_n; in the bottom, total difference between target and model for σ_d = 0.1 (dotted line), σ_d = 0.01 (dashed line), σ_d = 0.001 (dot-dashed line). Middle row (**b**,**e**): optimized approach. Bottom row (**c**,**f**): random parameter setting. The solid lines represent dedicated decay powers.

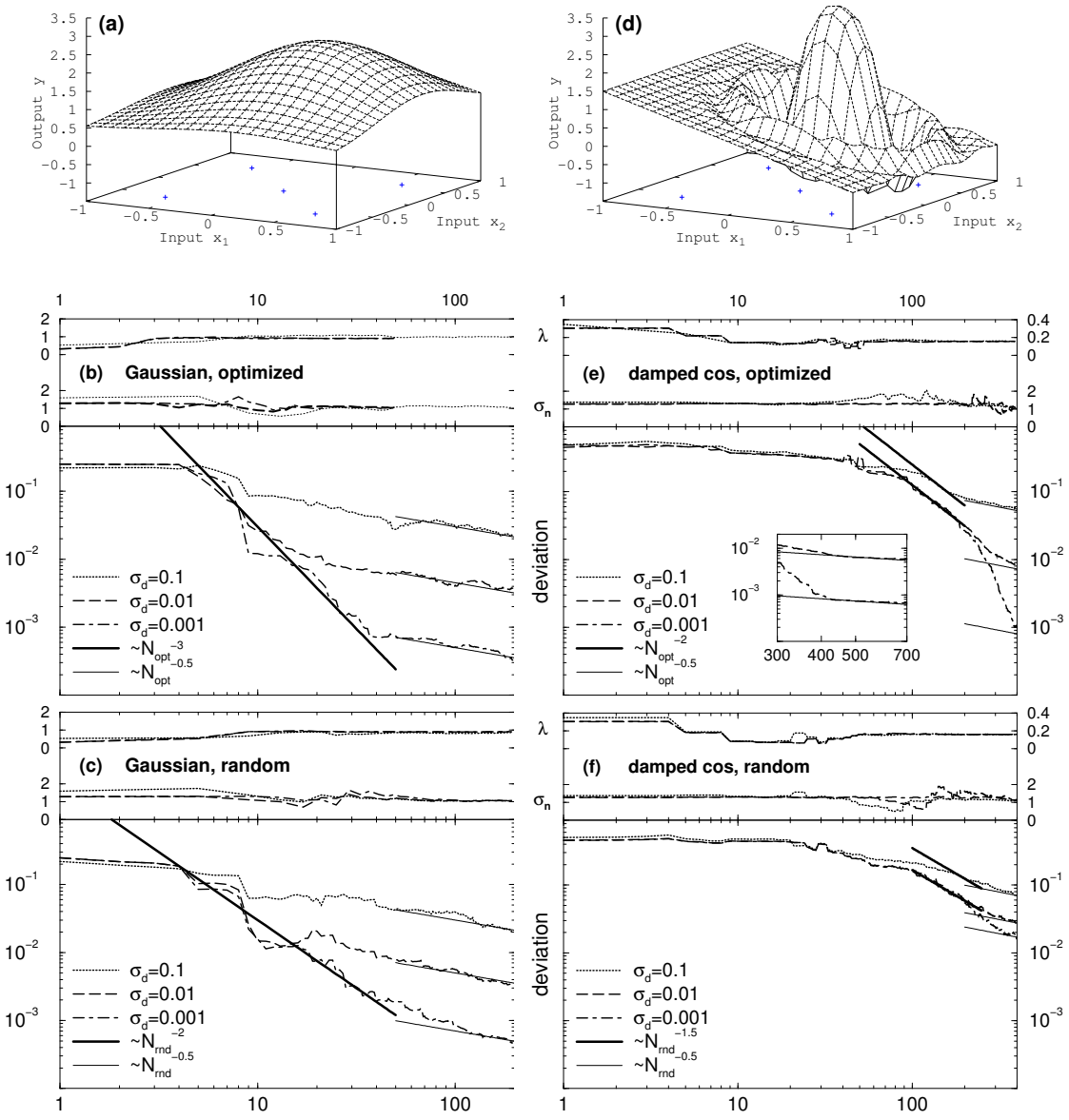

Figure 4. Two test cases for two-dimensional input vectors, $N_{\mathrm{ROI}} = 21 \times 21$ (ROI: region of interest). (**a**–**c**): Gaussian model; (**d**–**f**): damped cosine model. Top row (**a**,**d**): model, five initial input vectors (plus signs in base). Panels (**b**,**c**,**e**,**f**) with number of added points to the right: in the top of each figure, hyper-parameters λ and σ_n; in the bottom, total difference between target and model for $\sigma_d = 0.1$ (dotted line), $\sigma_d = 0.01$ (dashed line), $\sigma_d = 0.001$ (dot-dashed line). Middle row (**b**,**e**): optimized approach. Bottom row (**c**,**f**): random parameter setting. The solid lines represent dedicated decay powers. The small inset in (**e**) shows the deviations of target and model for $\sigma_d = 0.01$ and $\sigma_d = 0.001$, which settles on a square root behavior around 400 points.

The two-dimensional test cases in Figure 4 confirm the findings above. Since the small "waves" surrounding the large hump at the center of the damped cosine (Figure 4d) are much harder to resolve in two dimensions than is the case for the broad structure of the Gaussian (Figure 4a), the settlement of the hyper-parameters is shifted to higher numbers of acquired data. Again, the optimized approach surpasses the random choice.

8. Summary and Conclusions

A marginalization procedure was proposed to obtain different starting conditions for the batch execution of Gaussian processes in order to exploit parallel computing power. By investigating a one-dimensional test case, the implementation of the procedure (algorithm, computer program, Markov chain Monte Carlo results) was validated with analytic calculations. A marginalized input point was shown to become insignificant during the calculation procedure, thereby demonstrating the marginalization procedure being not harmful for the final result. Eventually, the convergence behaviour of the total difference of target and model was studied. It turned out that the result settles when the decay resembles a square root power law. Therefore, the onset of this behaviour may serve as a stopping criterion for the need of acquiring additional data points in the region of interest.

Acknowledgments: This work has been carried out within the framework of the EUROfusion Consortium and has received funding from the Euratom research and training programme 2014–2018 under grant agreement No. 633053. The views and opinions expressed herein do not necessarily reflect those of the European Commission.

Author Contributions: All authors contributed substantially to each step of the work. All authors have read and approved the final manuscript.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A. Notation Table

N	number of input data vectors
N_{dim}	number of elements in the input data vector
N_{marg}	number of marginalized test data
N_{opt}	number of points added to the data pool by optimization
N_{rnd}	number of points added to the data pool chosen randomly
N_{ROI}	number of test points in the region of interest
$\boldsymbol{x}_*$	test input vector
$\hat{\boldsymbol{x}}_1$	first test input vector, for which the target value is marginalized
$\boldsymbol{x}_{\mathrm{opt}}$	test input vector found by the utility criterion
$\boldsymbol{x}_{1*}$	test input vector after first marginalized test input vector was found
$\boldsymbol{x}_i = (x_{i1}, ..., x_{iN_{\mathrm{dim}}})$	i-th input data vector
$\boldsymbol{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_N)$	$N \times N_{dim}$ matrix with input data vectors as columns
$\boldsymbol{X}' = \{\boldsymbol{X}, \boldsymbol{\xi}\}$	matrix of the input data vectors expanded by the vector of grid points
$\boldsymbol{\xi}$	vector of grid points within region of interest $\mathcal{I}$
$\boldsymbol{\xi}_{\mathrm{max}}$	grid point with largest utility
f_*	target value at test input vector $\boldsymbol{x}_*$
$f(\boldsymbol{x})$	function of input data to describe target data
f_1	first target value, to be marginalized
f_{1*}	target value at test point, obtained after marginalization of a first target value
$\boldsymbol{y} = (y_1, ..., y_N)^T$	vector of the N target data
ϵ	uncertainty of the target data
$\sigma_{d_i}^2$	variance of the i-th target data
$\boldsymbol{\Delta}_{ij} = \sigma_{d_i}^2 \delta_{ij}$	ij-th element of the $N \times N$ matrix of the variances of target data
λ	length scale to set up the notion of distance between input data vectors
σ_f^2	signal variance of the distribution over functions f
σ_n^2	overall noise in the data
$\boldsymbol{\theta} = (\lambda, \sigma_f, \sigma_n)$	vector of the hyper-parameters
$k(\boldsymbol{x}_p, \boldsymbol{x}_q)$	covariance of two input data vectors
$(\boldsymbol{k}_*)_i = k(\boldsymbol{x}_*, \boldsymbol{x}_i)$	short notation for the i-th element of the vector of covariances between test input vector and input data vector
$\boldsymbol{K}_{pq} = k(\boldsymbol{x}_p, \boldsymbol{x}_q)$	pq-th element of the $N \times N$ covariance matrix of the input data vectors
$\boldsymbol{K}'$	$(N+1) \times (N+1)$ covariance matrix of the expanded input $\boldsymbol{X}'$
$\mathcal{I}$	region of interest to run Gaussian processes
$U(\boldsymbol{\xi})$	utility of a target data obtained at input vector ξ

Appendix B. Algorithm for Computer Simulation

We briefly describe an algorithm for a computer simulation employing the batch approach introduced. The field of application will be the prediction of particle transport and plasma-wall interaction in the scrape-off layer in fusion plasma experiments. Up to now, a data base with the outcome for about 1500 parameter settings has been established [16]. The usual running time for obtaining the outcome for one set of parameters is on the order of months, so the particular setups have to be chosen well.

1. Compose input data vector from data base;
2. Set up batch run with N_{proc} processors;
3. Processor #1: code running without any marginalized point;
   Processor #$(i+1)$: code running for i marginalized points;
4. Return outcome; i.e., N_{proc} most promising parameter settings for the long running simulation code ready for batch execution.

References

1. Barber, D. *Bayesian Reasoning and Machine Learning*; Cambridge University Press: Cambridge, UK, 2012.
2. Bishop, C. *Neural Networks for Pattern Recognition*; Oxford University Press: Oxford, UK, 1996.
3. MacKay, D.J.C. *Information Theory, Inference, and Learning Algorithms; Cambridge University Press:* Cambridge, UK, 2003.
4. Cohn, D. Neural Network Exploration Using Optimal Experiment Design. *Neural Netw.* **1996**, *9*, 1071–1083.
5. Managing Uncertainty in Complex Models. Available online: http://www.mucm.ac.uk/Pages/Dissemination/RelatedPapers.html (accessed on 18 February 2017).
6. Seo, S.; Wallat, M.; Graepel, T.; Obermayer, K. Gaussian process regression: Active data selection and test point rejection. In *Mustererkennung 2000*; Springer: Berlin/Heidelberg, Germany, 2000; pp. 241–246.
7. Gramacy, R.B.; Lee, H.K.H. Adaptive Design and Analysis of Supercomputer Experiments. *Technometrics* **2009**, *51*, 130–145.
8. Mockus, J. *Bayesian Approach to Global Optimization*; Springer: Berlin/Heidelberg, Germany, 1989.
9. Sacks, J.; Welch, W.; Mitchell, T.; Wynn, H. Design and Analysis of Computer Experiments. *Stat. Sci.* **1989**, *4*, 409–435.
10. Locatelli, M. Bayesian Algorithms for One-Dimensional Global Optimization. *J. Glob. Optim.* **1997**, *10*, 57–76.
11. Azimi, J.; Fern, A.; Fern, X. Batch Bayesian Optimization via Simulation Matching. In *Advances in Neural Information Processing Systems 23*; Lafferty, J.D., Williams, C.K.I., Shawe-Taylor, J., Zemel, R.S., Culotta, A., Eds.; Curran Associates: Red Hook, NY, USA, 2010; pp. 109–117.
12. Azimi, J.; Jalali, A.; Fern, X. Hybrid Batch Bayesian Optimization. In Proceedings of the 29th International Conference on Machine Learning, Edinburgh, Scotland, 26 June–1 July 2012.
13. Gonzalez, J.; Osborne, M.; Lawrence, N. GLASSES: Relieving The Myopia of Bayesian Optimisation. *J. Mach. Learn. Res.* **2016**, *51*, 790–799.
14. Rasmussen, C.; Williams, C. *Gaussian Processes for Machine Learning*; MIT Press: Cambridge, UK, 2006.
15. Preuss, R.; von Toussaint, U. Gaussian Processes for SOLPS Data Emulation. *Fusion Sci. Technol.* **2016**, *69*, 605–610.
16. Coster, D. (IPP, Garching, Germany). Private communication, 2014.