Data Driven Analytics (Machine Learning) for System Characterization, Diagnostics and Control Optimization

Jinkyoo Park[1], Max Ferguson[2] and Kincho H. Law[2]

[1] Korea Advanced Institute of Science and Technology (KAIST), Korea
[2] Stanford University, Stanford, CA 94305, USA
law@stanford.edu

Abstract. This presentation discusses the potential use of machine learning techniques to build data-driven models to characterize an engineering system for performance assessment, diagnostic analysis and control optimization. Focusing on the Gaussian Process modeling approach, engineering applications on constructing predictive models for energy consumption analysis and tool condition monitoring of a milling machine tool are presented. Furthermore, a cooperative control optimization approach for maximizing wind farm power production by combining Gaussian Process modeling with Bayesian Optimization is discussed.

Keywords: Machine Learning, Predictive Models, Data-Driven Optimization

1 Introduction

The last decade has seen an increasing number of research and applications of machine learning [1]. As sensor technologies, data acquisition systems, and data analytics continue to improve, companies can now effectively and efficiently collect large, rapid, and diverse volumes of data and get valuable insights from the data. The availability of large volume of data has given strong impetus to data-driven decision making. There have been a growing interest in applying machine learning techniques to draw insights gained from the data in engineering [2]. These data-driven approaches are able to find highly complex and non-linear patterns in data of different types and transform the raw data into useful models. For instance, in the manufacturing domain, machine learning techniques have been applied for prediction [3,4], detection and classification [5,6], or forecasting [7] for problems of specific interests.

Engineering systems are inherently dynamic, uncertain, and complex. Majority of machine learning techniques provide point predictions that do not usually consider uncertainties in the models. As prediction and forecasting of future consequences carry many unknowns and uncertainties, a prediction model should provide some quantification of uncertainty for informed decision making [8,9,10]. In this paper, we discuss methods that are based on Bayesian statistic inference and illustrate their potential applications for performance characterization, condition diagnostic and control optimization of physical systems.

Gaussian Process Regression (GPR) is one popular approach that provides confidence bounds and distribution for predictive estimation. Both features are valuable in establishing the foundation for uncertainty quantification analysis. Without predefining the basis functions, the nonparametric GPR has been widely used for approximating a target function that represents the complex input and output relationships based on the observed data and predicts a target output with quantified uncertainty [11]. Note that GPR can also be used for classification by converting the regression output to class probability [12]. Due to its ability to quantify uncertainty in the predictive model, GPR has received increased attention in the machine-learning community. The GPR algorithm has been applied to many fields, including manufacturing, robotics, music rendering, and others [13-16]. In this paper, we discuss the use of GPR for system characterization and diagnostics, using a manufacturing milling machine tool as illustrative testbed application.

Machine learning has been broaden from building predictive and forecasting models to supporting optimization and decision making problems [17]. For many complex physical systems, constructing the analytical model and the objective function for optimal decision making can be difficult. One alternative would be to establish a model using the data collected from the physical system. A data-driven approach would involve using the sampled data to construct the model while searching for the optimal value that maximizes the objective target function at the same time. For implementation in a physical environment, the number of sampled trials should be kept to a minimum as each trial would likely involve the execution of some physical actions. Using Gaussian Process (GP) as a way to approximate the target function, Bayesian Optimization (BO) has been shown effective in finding the optimal values of a target function with a small number of sampled trials. [18]. For efficient sampling, BO uses an acquisition function that takes advantage of the estimated probability distribution of the target function so that the number of function evaluations (i.e. executions of the physical actions) is kept small. In other words, BO algorithm iteratively approximates the input and the output relationship of a target system using Gaussian Process (GP) regression and utilizes an acquisition function with the learned model to determine the trial inputs that can potentially improve the target values [18-20]. Bayesian optimization has been applied to problems such as the multi-armed bandit problems and sensor networks [21-24]. When implementing with a physical system, the trial actions are typically constrained by the physical limitation of the system. Furthermore, it may not be desirable to impose abrupt changes in successive actions. We propose a Bayesian Ascent algorithm which augments BO with trust region constraints to ensure successive actions do not deviate significantly and to avoid abrupt changes [25]. In this paper, we use the total power production of a wind farm with multiple wind turbines as an illustrative example for the execution of the data-driven Bayesian Ascent method.

The purpose of this paper is to illustrate the potential applications and implementations of Bayesian-based machine learning approach in engineering system, including performance characterization, condition diagnostics and control optimization. The paper is organized as follows: Section 2 provides a brief description of the Gaussian Process Regression method with illustrative examples involving the development of an energy prediction model and a tool condition diagnostic model for a milling machine

tool. Section 3 discusses Bayesian Optimization and its adoption for the wind farm power production problem. Section 4 summarizes the paper with a brief discussion.

2 System Characterization and Diagnosis with GPR

This section discusses the use of input and output data of a system to characterize the performance of the system and to perform diagnostic analysis. Specifically, Gaussian Process Regression (GPR) is employed to build predictive models for energy performance and tool conditions of a milling machine tool. Prior to the training process, the raw data collected are pre-processed to extract features that are deemed useful for the construction and execution of the predictive models. Using the feature data as training data points, GPR then approximates a target function that represents the input and output relationships without predefining a set of basis functions. Given a new input, the approximated target function is then used to predict the target output with uncertainty quantification. One desirable property of a GPR model is its ability to capture complex input and output relationships with a small number of hyper-parameters. Moreover, a GPR model can be trained with relatively small number of experimental training data sets but is able to give reasonable predictive estimation quantified with uncertainty measures.

2.1 Gaussian Process Regression (GPR)

Given a data set $\boldsymbol{D}^n = \{(\boldsymbol{x}^i, y^i) | i = 1, \dots, n\}$ with n samples, where $\boldsymbol{x}^{1:n} = (\boldsymbol{x}^1, \dots, \boldsymbol{x}^n)$ and $\boldsymbol{y}^{1:n} = (y^1, \dots, y^n)$ denote, respectively, the inputs and the corresponding (possibly noisy) output observations of the target function values $\boldsymbol{f}^{1:n} = \big(f(\boldsymbol{x}^1), \dots, f(\boldsymbol{x}^n)\big)$, Gaussian Process Regression (GPR) constructs a posterior distribution $p(f^{new}|\boldsymbol{D}^n)$ on the function value $f^{new} = f(\boldsymbol{x}^{new})$ corresponding to an unseen input $\boldsymbol{x}^{new}$. The prediction is given as a probability distribution rather than a point estimate. The probability distribution enables quantifying uncertainty in the target value. A detailed description of GPR can be found in [11,12]. The following summarize the basic steps for training a GPR model and performing prediction using the trained GPR model.

Gaussian Process Prior on Target Function Values. GPR uses Gaussian Process (GP) as a prior to describe the distribution on the target function $f(\boldsymbol{x})$. GP is defined as a collection of random variables, any finite number of which is assumed to be jointly Gaussian distributed. The distribution over the target function $f(\boldsymbol{x})$ can be fully described by its mean function $m(\boldsymbol{x}) = \mathrm{E}[f(\boldsymbol{x})]$ and a kernel function $k(\boldsymbol{x}, \boldsymbol{x}')$ that approximates the covariance $\mathrm{E}[(f(\boldsymbol{x}) - m(\boldsymbol{x}))(f(\boldsymbol{x}') - m(\boldsymbol{x}'))]$. That is, the prior on the function values is represented as:

$$p(\boldsymbol{f}^{1:n}) = GP\big(m(\cdot), k(\cdot,\cdot)\big) \tag{1}$$

where $m(\cdot)$ is a mean function capturing the overall trend in the target function value, and $k(\cdot,\cdot)$ is a kernel function used to approximate the covariance. The kernel (covariance) function represents a geometrical distance measure assuming that the more

closely located inputs would be more correlated in terms of their function values.

Defining Covariance (Kernal) Function. In GPR, the kernel (covariance) function describes the structure of the target function. A wide variety of kernel functions have been proposed and described in the literature [12,26]. The covariance kernel function provides an efficient method to estimate the correlation between two input feature vectors, $\boldsymbol{x}^i$ and $\boldsymbol{x}^j$. The type of kernel function chosen can strongly affect the representability of the GPR model, and influence the accuracy of the predictions. One common choice for a GPR model is the stationary and (infinitely) differentiable *squared exponential* (SE) kernel:

$$k_{SE}(\boldsymbol{x}^i, \boldsymbol{x}^j) = \sigma^2 \exp\left(-\frac{1}{2\lambda^2}\left\|\boldsymbol{x}^i - \boldsymbol{x}^j\right\|^2\right), \tag{2}$$

where the kernel function is described by the hyperparameters, σ and λ. The hyperparameter λ is commonly referred to as the length scale. The length scale quantifies whether two points at a certain distance apart in the input space are considered close together. The signal variance σ^2 quantifies the overall magnitude of the covariance value. While the SE kernel function is a good choice for many applications for a feature vector with multiple variables (or dimensions), it does not allow the length scale to vary for each dimension in the feature vector. A common alternative is to use an *automatic relevance determination* (ARD) kernel, which assigns a different length scale to each dimension. The ARD squared exponential (ARD-SE) kernel, which is essentially a product of the SE kernels over different dimensions, can be expressed as:

$$k_{ARD-SE}(\boldsymbol{x}^i, \boldsymbol{x}^j) = \sigma^2 \exp\left(-\frac{1}{2}(\boldsymbol{x}^i - \boldsymbol{x}^j)^T \mathrm{diag}(\boldsymbol{\lambda})^{-2}(\boldsymbol{x}^i - \boldsymbol{x}^j)\right). \tag{3}$$

An ARD-SE kernel provides the flexibility to adjust the relevance (weight) of each parameter in the feature vector where the parameter vector $\boldsymbol{\lambda} = (\lambda_1, \ldots \lambda_i, \ldots, \lambda_m)$ is referred to as the characteristic length scales that quantify the relevancy of the input features in $\boldsymbol{x^i} = \left(x_1^i, \ldots x_k^i, \ldots, x_m^i\right)$, where *m* defines the number of input variables, for predicting the response y. A large length scale λ_i indicates weak relevance for the corresponding input feature $\boldsymbol{x}^i$and vice versa.

Other kernel functions such as linear, periodic, Matern kernals exist [12,26,27]. New kernel function can be constructed by combining kernal functions via, for examples, addition and multiplication of the kernel functions [12,26]. The kernel function produces a positive semi-definite symmetric kernel matrix $\mathbf{K}$ whose (i,j)th entry is $\mathbf{K}_{ij} = k(\boldsymbol{x}^i, \boldsymbol{x}^j)$.

Defining Likelihood Function (Observation Model). The latent random variables $\boldsymbol{f}^{1:n}$ needs to be inferred from the observations $\boldsymbol{y}^{1:n} = (y^1, \ldots, y^n)$. Each observed (or response) value is assumed to contain some random noise ϵ^i, such that $y^i = f(\boldsymbol{x}^i) + \epsilon^i$. Different likelihood functions can be used to model the random noise term. It is common to assume that the noise term $\epsilon^i \sim \mathcal{N}(0, \sigma_\epsilon^2)$ is independent and identically distributed Gaussian with zero mean, in which case the likelihood function becomes:

$$p(\boldsymbol{y}^{1:n}|\boldsymbol{f}^{1:n}) = \mathcal{N}(\boldsymbol{f}^{1:n}, \sigma_\epsilon^2 \mathbf{I}) \tag{4}$$

where the noise variance, σ_ϵ^2, quantifies the level of noise that exists in the observations. The Gaussian likelihood function is used because it guarantees that the posterior can also be expressed as a Gaussian distribution.

Training the Regression Model (Optimizing Hyper-Parameters). Including the noise model, the covariance function is parameterized (i.e., defined) by the hyperparameters jointly denoted by $\boldsymbol{\theta} = (\sigma_\epsilon, \sigma, \boldsymbol{\lambda})$.The marginal distribution of the observations (conditioned on the hyperparameters) can be expressed as:

$$p(\boldsymbol{y}^{1:n}|\boldsymbol{\theta}) = \int p(\boldsymbol{y}^{1:n}|\boldsymbol{f}^{1:n}, \boldsymbol{\theta})\, p(\boldsymbol{f}^{1:n}|\boldsymbol{\theta})\, d\boldsymbol{f}^{1:n} \ \left(= \mathcal{N}(0, \mathbf{K} + \sigma_\epsilon^2 \mathbf{I})\right) \tag{5}$$

Note that since the GP prior and Gaussian likelihood function are used, the marginal distribution is also Gaussian. One attractive property of the GP model is that it provides an analytical closed form expression for the marginal likelihood of the data (with the unknown latent function $f(\boldsymbol{x})$ being "marginalized" out). The marginal log-likelihood for the training data set $\boldsymbol{D}^n = \{(\boldsymbol{x}^i, y^i)|i = 1, \dots, n\}$ can be expressed as [11,12]:

$$\log p\,(\boldsymbol{y}^{1:n}|\boldsymbol{\theta}\,) = -\frac{1}{2}(\boldsymbol{y}^{1:n})^T(\mathbf{K} + \sigma_\epsilon^2\mathbf{I})^{-1}\boldsymbol{y}^{1:n} - \frac{1}{2}\log|\mathbf{K} + \sigma_\epsilon^2\mathbf{I}| - \frac{n}{2}\log 2\pi \tag{6}$$

The hyperparameters $\boldsymbol{\theta} = (\sigma_\epsilon, \sigma, \boldsymbol{\lambda})$ for the noise model and the kernel function are determined as ones maximizing the marginal log-likelihood of the training data $\boldsymbol{D}^n = \{(\boldsymbol{x}^i, y^i)|i = 1, \dots, n\}$ as [12]:

$$\begin{aligned} \boldsymbol{\theta}^* &= \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \log p\,(\boldsymbol{y}^{1:n}|\boldsymbol{\theta}\,) \\ &= \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \left(-\frac{1}{2}(\boldsymbol{y}^{1:n})^T(\mathbf{K} + \sigma_\epsilon^2\mathbf{I})^{-1}\boldsymbol{y}^{1:n} - \frac{1}{2}\log|\mathbf{K} + \sigma_\epsilon^2\mathbf{I}| - \frac{n}{2}\log 2\pi\right) \end{aligned} \tag{7}$$

As long as the kernel function is differentiable with respect to its hyper-parameters $\boldsymbol{\theta}$, the marginal likelihood can be differentiated and optimized by various off-the-shelf mathematical programming tools, such as GPML [28], scikit-learn [29], and others.

Constructing Posterior Predictive Distribution (Perdiction). Once the optimized hyperparameters are obtained, the trained GPR model is fully characterized. Let's denote a newly observed input $\boldsymbol{x}^{new}$:

$$\boldsymbol{x}^{new} = (x_1^{new}, \dots x_k^{new}, \dots x_m^{new})$$

The (hidden) function value $f(\boldsymbol{x}^{new})$ for the new input $\boldsymbol{x}^{new}$ and the observed outputs $\boldsymbol{y}^{1:n} = \{y^1, \dots, y^n\}$ follow a multivariate Gaussian distribution:

$$\begin{bmatrix} \boldsymbol{y}^{1:n} \\ f(\boldsymbol{x}^{new}) \end{bmatrix} \sim N\left(\mathbf{0}, \begin{bmatrix} (\mathbf{K} + \sigma_\epsilon^2\mathbf{I}) & \boldsymbol{k} \\ \boldsymbol{k}^T & k(\boldsymbol{x}^{new}, \boldsymbol{x}^{new}) \end{bmatrix}\right) \tag{8}$$

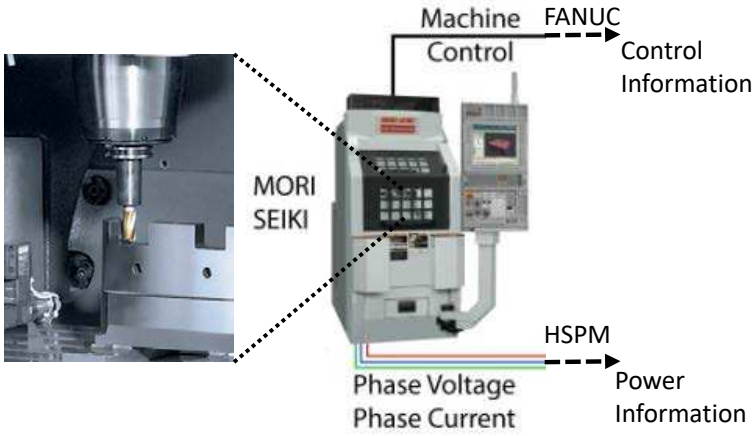


Fig. 1. A Mori Seiki NVD 1500DCG 3-axis milling machine

where $\boldsymbol{k}^T = \left(k(\boldsymbol{x}^1, \boldsymbol{x}^{new}), \ldots, k(\boldsymbol{x}^n, \boldsymbol{x}^{new})\right)$. The posterior predictive distribution on the response $f(\boldsymbol{x}^{new})$ for the newly observed (and previously unseen) input $\boldsymbol{x}^{new}$ given the historical data $\boldsymbol{D}^n = \{(\boldsymbol{x}^i, y^i) | i = 1, \ldots, n\}$ can then be expressed as a 1-D Gaussian distribution $f(\boldsymbol{x}^{new}) \sim N\left(\mu(\boldsymbol{x}^{new}|\boldsymbol{D}^n), \sigma^2(\boldsymbol{x}^{new}|\boldsymbol{D}^n)\right)$ with the mean and variance functions expressed, respectively, as [11,12]:

$$\mu(\boldsymbol{x}^{new}|\boldsymbol{D}^n) = \boldsymbol{k}^T(\mathbf{K} + \sigma_\epsilon^2\mathbf{I})^{-1}\boldsymbol{y}^{1:n} \tag{9}$$

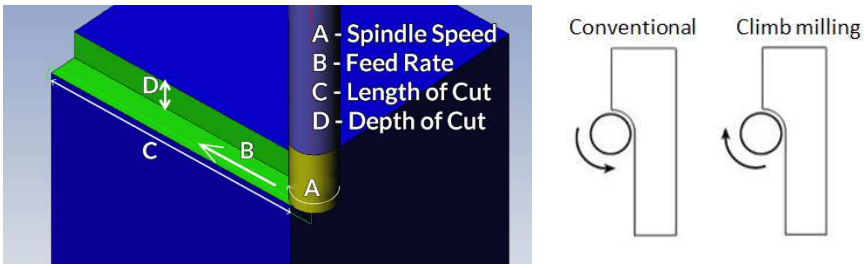
$$\sigma^2(\boldsymbol{x}^{new}|\boldsymbol{D}^n) = k(\boldsymbol{x}^{new}, \boldsymbol{x}^{new}) - \boldsymbol{k}^T(\mathbf{K} + \sigma_\epsilon^2\mathbf{I})^{-1}\boldsymbol{k} \tag{10}$$

Here, $\mu(\boldsymbol{x}^{new}|\boldsymbol{D}^n)$ and $\sigma^2(\boldsymbol{x}^{new}|\boldsymbol{D}^n)$ can be used as the scoring functions for evaluating, respectively, the mean and variance of the hidden function output $f(\boldsymbol{x}^{new})$ corresponding to the input data $\boldsymbol{x}^{new}$.

2.2 Characterizing Energy Consumption of a Milling Machine

Monitoring and optimizing energy efficiency of the manufacturing processes has become a priority in the manufacturing industry. This section discusses how a GPR-based energy prediction model for a milling machine is established. An energy prediction model can provide a better understanding of how different operational strategies may influence the energy consumption pattern of a machine tool and enable selection of optimal strategy with efficient operations for machining a part.

Figure 1 shows the basic set up of a Mori Seiki NVD 1500DCG 3-axis milling machine tool. The machining process data, such as process parameters, NC blocks, and tool positions, are collected from the FANUC controller and the power time series data is collected using a High Speed Power Meter (HSPM). With recent technologies and standards, such as MTConnect [30], it is now possible to track variations in energy consumption by different machine operations [31]. The raw data collected includes the



(a) Process parameters of a milling process (b) Cutting strategy

Fig. 2. Machining process parameters

timestamp, time series data for power consumption, feed rate and spindle speed, and numerical control (NC) code information. The raw data is then post-processed to derive the information such as average feed rate, average spindle speed, cumulative energy consumption, volume of material removed, depth of cut and cutting strategy for each NC code block. The hardware platform, the data acquisition system, the experimental design, and the data processing techniques have been described in details by Bhinge et al. [32].

In the example presented herein, the input process parameters used are shown in Figure 2. The input features employed are defined as follows:

- $x_1 \in \mathbb{R}$ Feed rate: the velocity at which the tool is fed
- $x_2 \in \mathbb{R}$ Spindle speed: rotational speed of the tool
- $x_3 \in \mathbb{R}$ Depth of cut: depth of material that the tool is removing
- $x_4 \in \{1, 2, 3, 4\}$ Active cutting direction: (1 is for x-axis, 2 for y-axis, 3 for z-axis, and 4 for x-y axes)
- $x_5 \in \{1, 2, 3\}$ Cutting strategy: the method for removing material (1 is for conventional, 2 for climbing, and 3 for both)

Each operation, i.e. a feature data, refers to a single NC block. The energy consumption, $E \in \mathbb{R}$, for each NC block is obtained by numerically integrating the power time series recorded by the HSPM over the duration of the NC block. Additionally, the length of the tool path. $l \in \mathbb{R}$, in a single NC block is computed using the lengths of cut in the x-, y- and z-directions. The output parameter, y, is defined as the energy consumption per unit length (i.e. $y^i = E^i/l^i$ for the i^{th} NC block operation).

For this example, the data for a total of 196 face milling experiments for machining parts using the milling machine are collected. We choose to use all of the measured input features, and define the input feature vector, $\boldsymbol{x} = \{x_1, \dots, x_5\}$. We assume that the output measurement $y = f(\boldsymbol{x}) + \epsilon$ contains noise $\epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2)$. The ARD-SE kernel is used as the covariance function. Furthermore, we assume the mean function from the prior (see Eq. (1)) to be a zero function.

The GPR model is used to generate energy density predictions $\hat{y}^i$ for a new set of operating conditions $\boldsymbol{x}^i$. Let $\boldsymbol{D^{train}}$ denote the training data set containing the input feature data and the output parameter data. Each new prediction $\hat{y}^i$ is represented by a mean energy density function $\mu(\boldsymbol{x}^i|\boldsymbol{D^{train}})$ and associated standard deviation function

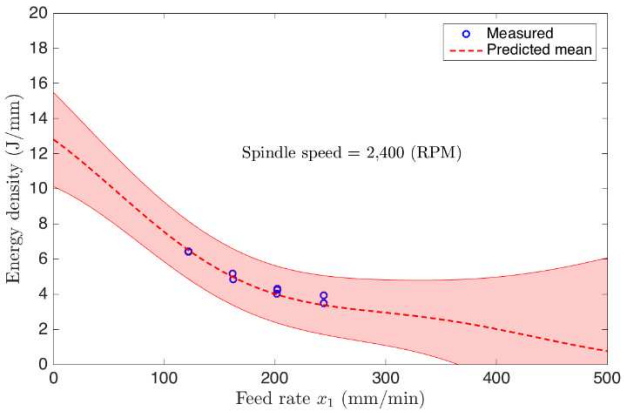


Fig. 3. Predicted energy consumption density for generic test parts machined using face milling. The operating parameters include y-direction cut, 2,400 RPM spindle speed, conventional cutting strategy, and 1mm cut depth. The shaded area shows one-standard deviation bounds for the prediction.

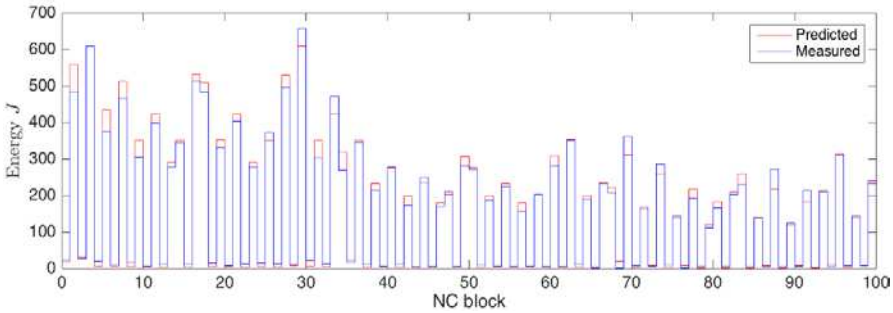


Fig. 4. Predicted mean energy consumption for face milling operations using the GPR model. Results are compared to the actual energy consumption of the machine.

$\sigma(\boldsymbol{x}^i|\boldsymbol{D}^{\boldsymbol{train}})$. We can then estimate the energy consumption $\hat{E}^i$ and the corresponding standard deviation S^i on the estimated energy consumption value as:

$$\hat{E}^i = \mu(\boldsymbol{x}^i|\boldsymbol{D}^{\boldsymbol{train}}) \times l_i \tag{11}$$

$$S^i = \sigma(\boldsymbol{x}^i|\boldsymbol{D}^{\boldsymbol{train}}) \times l_i \tag{12}$$

Details on developing the GPR model have been reported in [14]. Figure 3 shows the energy consumption predicted by the GPR model. As shown in Figure 4, the GPR model provides a good estimation of the energy consumption of the milling machine on the test data set.

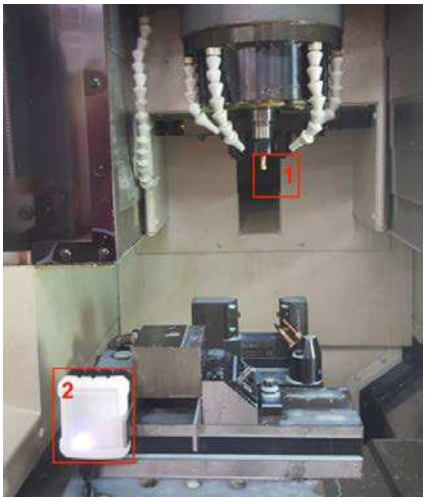
This example illustrates that the GPR models can be established to calculate the complex relationship between the input machining parameters and output energy consumption, and construct a prediction function for the energy consumption with confidence bounds. While this example uses face milling as a demonstrative example, the same technique has shown to be applicable to other milling operations [13]. Furthermore, since the input features are directly related to the NC code information (which can often be generated from a CAD model) the saved GPR model can be used to predict energy consumption when machining a new part as well to pre-determine the best tool path for machining a part with minimal energy [13,14].

2.3 Diagnosis of Tool Conditions

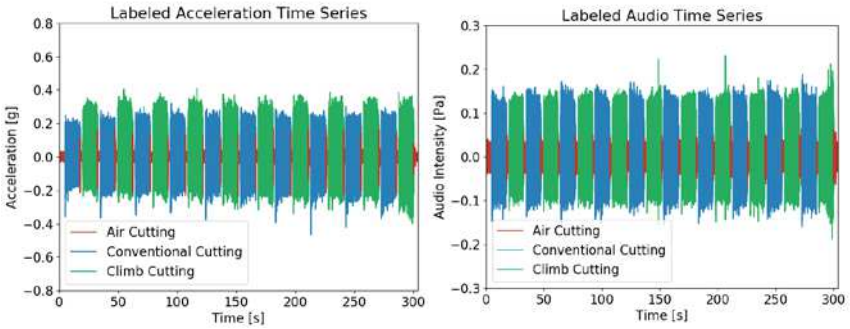
This section describes an application of GPR in developing a predictive model for tool condition diagnostic. Researchers have previously demonstrated that the condition of a machine tool can be inferred from features of the vibration and audio time series [33,34,35]. With the availability of low cost sensors, it is possible to collect real time vibration and audio data from critical locations inside a manufacturing machine tool. As shown in Figure 5, a waterproof sensor unit from Infinite Uptime, Inc. is attached to the vise of the Mori Seiki NVD 1500DCG 3-axis milling machine. The sensor unit is capable of measuring both the audio and triaxial acceleration signals inside the milling machine. In the experimental set up, the acceleration signal is recorded in the *x*-, *y*- and *z*-direction with a sampling rate of 1000 Hz to capture the 200 Hz signal generated by the cutting tool when the spindle rate is set to 3000 RPM. The audio signal is recorded at 8000 Hz. Data is streamed from the sensor to a laptop computer using a universal Serial Bus (USB) connection. Figure 5(b) shows the time series data in the production of a part that involves 10 *climb-cutting* operations and 10 *conventional-cutting* operations. Each cutting operation is separated by a brief *air-cutting* operation (without removing materials), in which the machine pauses briefly between cuts.

The milling machine was programmed to produce a number of simple parts by removing material from a solid steel block until the cutting tool became severely damaged, or the cutting tool broke. Tests were conducted using an Atrax solid carbide 4-flute square end mill, and a continuous supply of coolant. As described previously, the machining data, such as tool position and rotation speed, can be recorded from the FANUC controller and streamed to a laptop computer, along with a timestamp. The data is then post-processed to extract the behavior of the machine from the raw numerical control (NC) code [32].

In this work, we define the condition of the milling machine tool $y \in [0,1]$, based on the remaining lifetime of the tool, as estimated after manually examining the tool with a microscope. The scale is defined such that 1 indicates a new tool in perfect condition, and 0.5 indicates the condition at which the tool would be replaced in a commercial manufacturing operation as judged by a machine operator. Figure 6 shows examples of different levels of tool wear condition.

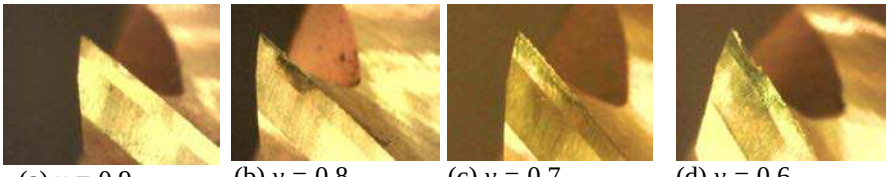


(a) Milling machine showing (1) the cutting tool and (2) the sensor unit

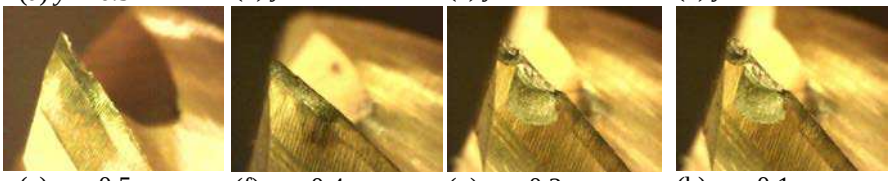


(b) Recorded vibration and acoustic time series signals

Fig. 5. Experimental Set Up for Tool Condition Diagnosis

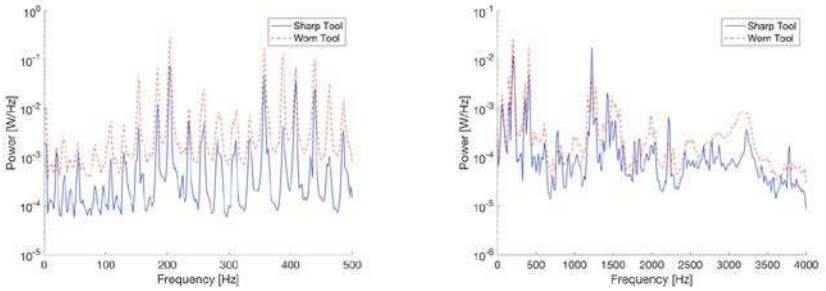


(a) y = 0.9 (b) y = 0.8 (c) y = 0.7 (d) y = 0.6



(e) y = 0.5 (f) y = 0.4 (g) y = 0.2 (h) y = 0.1

Fig. 6. Solid carbide end mill flute in different states of condition. A lower value of y indicates higher tool wear condition



(a) Acceleration periodograms (b) Audio periodograms

Fig. 7. Periodograms recorded with a sharp tool and a worn tool for a climb-cutting operation

As the audio and vibration signals are periodic, it seems natural to transform the time series signals into power spectra in the frequency domain. First, an attempt is made to data [36,37,38]. In Welch's method, a discrete time series signal s, is divided into K successive blocks s_m, using a window function w:

$$s_m(n) = w(n)s(n + mR), \quad n = 0 \dots M - 1, \qquad m = 0 \dots K - 1 \tag{13}$$

where M is the length of the window and R is the window hop size [37]. Furthermore, we use the Hann window function to reduce the spectral leakage [37]:

$$w(n) = \begin{cases} 0.5\left(1 - \cos\left(\dfrac{2\pi n}{M-1}\right)\right) & if \quad n \le M-1; \\ 0 & otherwise. \end{cases} \tag{14}$$

The periodogram of the m^{th} block is then calculated using the Fourier transform:

$$\boldsymbol{p}_m(\omega_k) = \frac{1}{M}\left|\sum_{n=0}^{N-1} s_m(n) e^{-\frac{i2\pi nk}{N}}\right|^2. \tag{15}$$

where ω_k is the k^{th} point in in the discretized frequency domain. The Welch estimate of the power spectral density is given by:

$$\hat{\boldsymbol{s}}(\omega_k) = \frac{1}{K}\sum_{m=0}^{K-1} \boldsymbol{p}_m(\omega_k). \tag{16}$$

The Welch's method essentially computes the average of periodograms across time. Each periodogram is obtained from a windowed segment of the time series. In this example case study, a window overlap of 50% is used and a Hann window length of 256 points is chosen for the vibration and audio signals. The length of the window is equal to the number of intervals in the discretized frequency domain. Figures 7(a) and 7(b) show respectively the periodograms for sample acceleration and audio signals produced by a sharp and a worn tool. It can be observed that the signal amplitudes tend to produce

larger amplitude when the tool is worn.

In the tool condition model, we choose to use the periodograms as input features which, as described, can be easily constructed in real time as data is collected. We denote the vibration periodogram $\hat{\boldsymbol{s}}_v^i \in \mathbb{R}^{256}$ and acoustic periodogram $\hat{\boldsymbol{s}}_a^i \in \mathbb{R}^{256}$ for each milling machine operation i. The condition of the milling tool tends to change gradually under normal operation. For this reason, we choose to include the previous condition of the milling tool in the feature vector. Altogether, we denote the input features for each milling operation i as:

$$\boldsymbol{x}^i = \begin{bmatrix} c^i \\ \hat{\boldsymbol{s}}_v^i \\ \hat{\boldsymbol{s}}_a^i \end{bmatrix}. \tag{17}$$

where c^i is the best estimate of the previous tool condition. During the training procedure, the previous tool condition is known, so c^i can be set equal to y^{i-1}:

$$c_{training}^i = \begin{cases} 1 & \text{for } i = 1, \\ y^{i-1} & \text{otherwise.} \end{cases} \tag{18}$$

To make a new prediction using the scoring procedure, the feature vector $\boldsymbol{x}^{new}$ first needs to be derived from the milling machine data. In a manufacturing setting, the vibration and audio periodograms can be calculated immediately after each operation is performed by the milling machine. However, the previous tool condition value y^{i-1}, will not be known. Therefore, we use the previous tool condition prediction, $\hat{y}^{i-1}$, in the scoring procedure, that is:

$$c_{testing}^i = \begin{cases} 1 & \text{for } i = 1, \\ \hat{y}^{i-1} & \text{otherwise.} \end{cases} \tag{19}$$

This makes the prediction process recursive. The first prediction is made by assuming a new tool, i.e. $c_{testing}^{i=1} = 1$. All subsequent predictions are made using the previous prediction as a starting point.

An ideal kernel for the tool condition model would allow the length scale of each periodogram to be varied independently, without introducing a large number of additional hyperparameters. When using the ARD squared exponential kernel, the number of hyperparameters grows linearly with the dimensional size of the feature vector, making the model prone to overfitting [39]. Instead, we choose to combine the SE kernels for each data type [40]. The resulting *sum of square exponential* (SSE) kernel is defined as follows:

$$\begin{aligned} k_{SSE}(\boldsymbol{x}^i, \boldsymbol{x}^j) = \sigma_1^2 \exp\left(-\frac{1}{2\lambda_1^2}\left\|c^i - c^j\right\|^2\right) + \sigma_2^2 \exp\left(-\frac{1}{2\lambda_2^2}\left\|\hat{\boldsymbol{s}}_v^i - \hat{\boldsymbol{s}}_v^j\right\|^2\right) \\ + \sigma_3^2 \exp\left(-\frac{1}{2\lambda_3^2}\left\|\hat{\boldsymbol{s}}_a^i - \hat{\boldsymbol{s}}_a^j\right\|^2\right), \end{aligned} \tag{20}$$

where $\sigma_1, \sigma_2, \sigma_3, \lambda_1, \lambda_2$ and λ_3 are the parameters to be determined for the SSE kernel function. Including the noise term σ_ϵ^2, the model with the SSE kernel function is described by seven hyperparameters, $\boldsymbol{\theta} = [\sigma_1, \sigma_2, \sigma_3, \lambda_1, \lambda_2, \lambda_3, \sigma_\epsilon]$. When compared to

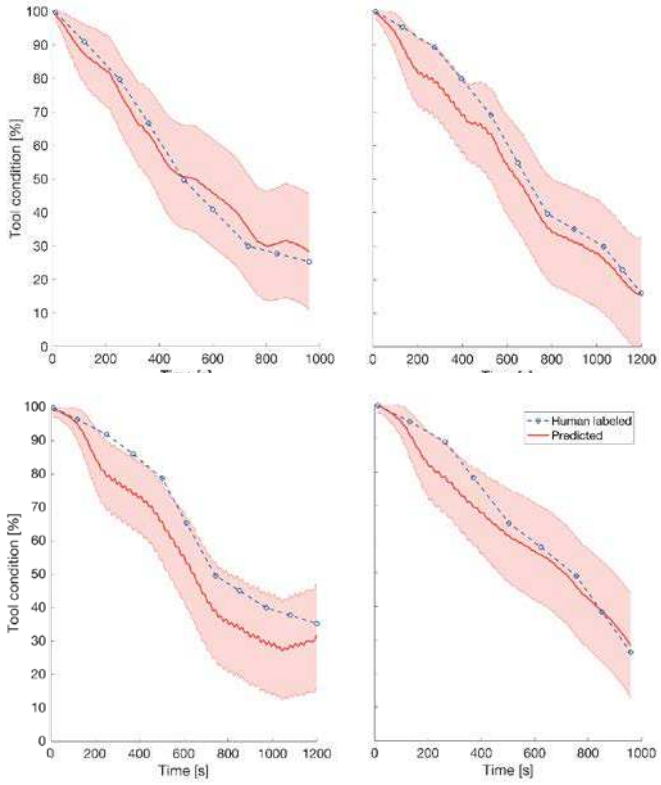


Fig. 8. Tool condition prediction for three testing data sets. Each plot represents a test where the milling machine was run until the cutting tool became severely worn or broken. The shaded region represents the 90% confidence interval for each prediction.

the ARD-SE kernel, the SSE kernel has fewer hyperparameters, but still allows the length scale of the previous state and periodograms to be adjusted independently.

We randomly select 14 experiments for the training set and 4 experiments for the testing set. Two GP models are trained to predict tool condition; the first is trained with climb-cutting data from the training set, and the second is trained with conventional-cutting data from the training set. The two models are then used together to predict the condition of the tool for each test case. As shown in Figure 8, even with a small number of training data sets, the predictive results give reasonable accurate prediction for good tool conditions. The figure also reveals that the confidence interval on the predicted value is large when the tool is worn. There are several reasons for this: firstly, the tool is more likely to break or undergo rapid condition change at this stage; hence, there is less certainty when predicting the condition of a heavily worn tool. Secondly, the training set contains less data for heavily worn tools, as some of the tests were ended abruptly when the tool broke. The reduced amount of training data for heavily worn tools

makes predictions less certain. Finally, the amplitudes of the acceleration signal generated by heavily worn tools have a much larger variance. For example, if a single flute on the tool is worn heavily, the tool is no longer symmetric so the corresponding acceleration signal will have a very high amplitude. On the other hand, if the tool is worn evenly the corresponding acceleration signal may have a much smaller amplitude. Many of these limitations may have been exacerbated by the accelerated nature of the tool wear experiment. It is important to note that tool condition would rarely be allowed to pass below 50% in a real manufacturing setting. Once the tool exceeds this level of wear, the surface quality of the part deteriorates quickly. This example illustrates the potential use of GPR as a means for predictive tool condition monitoring.

3 System Control Optimization

This section discusses real time control of a physical system, for which the construction of the analytical model and the objective function is difficult. The strategy is to iteratively select a series of trial input actions that potentially optimize the objective, observe the corresponding outputs, and learn about the unknown target function based on the inputs and the outputs. One important consideration for control of a physical system using a data-driven approach is to ensure that the number of trials is kept small as a trial would involve execution of corresponding control actions. Utilizing GPR to establish a learned model based on the input trials and output measurements, Bayesian Optimization (BO) has been found effective in optimizing a target function using a small number of trials. To illustrate, we discuss an application of BO to maximize the power production of a wind farm with multiple wind turbines in an experimental wind tunnel study.

3.1 Bayesian Optimization

Bayesian Optimization (BO) algorithm iteratively approximates the input and the output relationship of a target system using Gaussian Process (GP) regression and uses the learned model to determine the trial inputs that can potentially improve the target values [18,19,20]. Following the GPR procedure described in the previous section, denote $\boldsymbol{x} = (x_1, \dots, x_m)$ as the m input control actions and let $f(\boldsymbol{x})$ be the unknown target function for the output measurement y inferred by $\boldsymbol{x}$. The output measurement $y = f(\boldsymbol{x}) + \epsilon$, is assumed to include noise ϵ which follows a Gaussian distribution, i.e., $\epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2)$ [18,19,20]. At the nth iteration, using the historical input-output data $\boldsymbol{D}^n = \{(\boldsymbol{x}^1, y^1), \dots, (\boldsymbol{x}^i, y^i), \dots, (\boldsymbol{x}^n, y^n)\}$, where $\boldsymbol{x}^i = \left(x_1^i, \dots, x_m^i\right)$ and y^i represent, respectively, the input actions and the corresponding output measurement at the ith iteration, BO attempts to improve the target function by executing three basic steps:

- Learning: model the unknown target function $f(\boldsymbol{x})$ using Gaussian Process (GP) regression.
- Optimization: select the next trial input $\boldsymbol{x}^{n+1}$ that is useful for increasing (exploitation) and learning (exploration) the objective function $f(\boldsymbol{x}) \sim \mathcal{N}(\mu(\boldsymbol{x}|\boldsymbol{D}^n), \sigma^2(\boldsymbol{x}|\boldsymbol{D}^n))$.

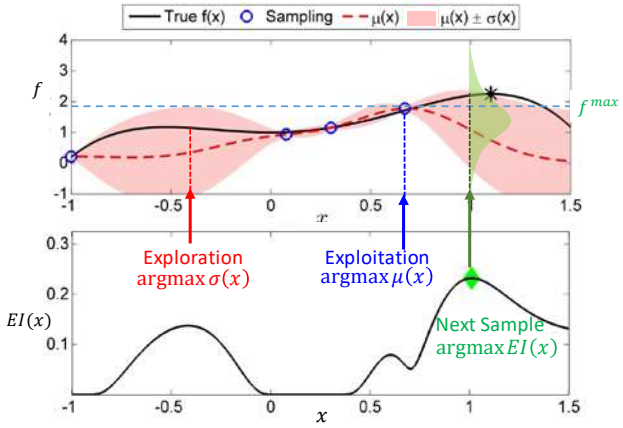


Fig. 9. Acquisition function incorporating exploration and exploitation

- Observation: execute the selected actions $\boldsymbol{x}^{n+1}$ and obtain the corresponding output y^{n+1} from the target system.

The new input and output pair $(\boldsymbol{x}^{n+1}, y^{n+1})$ will then be used to update the regression model for the target function $f(\boldsymbol{x})$ in the next iteration. The key of the BO algorithm lies in the optimization step for selecting a trial action to improve towards the optimal control actions.

As discussed, GPR is an effective method to establish a non-parametric regression model target function $f(\boldsymbol{x})$ to represent the complex input-output relationship of a system. The model is defined by a mean function $\mu(\boldsymbol{x}|\boldsymbol{D}^n)$ and a variance function $\sigma^2(\boldsymbol{x}|\boldsymbol{D}^n)$ obtained from the GP regression. Using the GPR model, the goal is to select the next input $\boldsymbol{x}^{n+1}$ in order to *learn* more about the target function as well as to *improve* the target value at the same time. The strategy would be to select the next input by exploiting the current belief about the target system by maximizing the current mean function $\mu(\boldsymbol{x}|\boldsymbol{D}^n)$ as well as exploring the uncertainty (variance) around the selected input by maximizing the variance function $\sigma^2(\boldsymbol{x}|\boldsymbol{D}^n)$. In general, the next sampling input is selected as the one that maximizes an acquisition function that incorporates both the aspects of exploration and exploitation as illustrated as shown in Figure 9. In this study, we select the next input $\boldsymbol{x}^{n+1}$ based on maximizing the expected improvement (EI) acquisition function as [41]:

$$\boldsymbol{x}^{n+1} = \arg\max_{\boldsymbol{x}\in\boldsymbol{A}\cap\boldsymbol{T}} \mathrm{EI}(x) \triangleq \arg\max_{\boldsymbol{x}\in\boldsymbol{A}\cap\boldsymbol{T}} \mathrm{E}[\max\{0, f(\boldsymbol{x}) - f^{max}\}|\boldsymbol{D}^n] \tag{21}$$

Here, $\boldsymbol{A} := \{\boldsymbol{x}|\ \boldsymbol{x}^l \le \boldsymbol{x} \le \boldsymbol{x}^u\}$ defines the ranges of allowable values for the actions $\boldsymbol{x}$, $\boldsymbol{T} := \{\boldsymbol{x}|\ \|x_i - x_i^{max}\| < \tau_i \text{ for } i = 1, \dots, m\}$ is a trust region used to restrict the change of actions, and $f^{max} = \max_{\{\boldsymbol{x}\in\boldsymbol{x}^{1:n}\}} \mu(\boldsymbol{x}|\boldsymbol{D}^n)$ is the maximum target function value estimated in the (current) nth iteration. The ranges for system constraints $\boldsymbol{A}$ define the limits of trial actions allowed by the physical system. For a physical system,

16

abruptly changing the control actions may not be desirable. The trust region $\boldsymbol{T}$ is imposed as a proximity constraint, τ_i, that limits the range of change allowable for the actions for each iterative step. We call the BO algorithm with the trust region constraint Bayesian Ascent (BA) method, in that the algorithm follows the ascending direction estimated probabilistically from a sequence of observations [25].

The quantity $\max\{0, f(\boldsymbol{x}) - f^{max}\}$ denotes the improvement toward the maximum output $f(\boldsymbol{x})$ with respect to the estimated maximum function value f^{max}. Here we use f^{max} instead of the actually observed maximum response y^{max} because the measurement value y^{max} may have large noise that could interfere with the sampling procedure. Given the estimated f^{max}, the value of the expected improvement $\mathrm{EI}(\boldsymbol{x})$ can be analytically derived using the distribution of the target function $f(\boldsymbol{x})$ at $\boldsymbol{x}$ [41]:

$$EI(x) = \begin{cases} (\mu(x) - f^{max})\Phi(Z) + \sigma(x)\phi(Z) & if\ \sigma(x) > 0 \\ 0 & if\ \sigma(x) = 0 \end{cases} \tag{7}$$

where $\phi(\cdot)$ and $\Phi(\cdot)$ denote, respectively, the probability and cumulative distribution functions, and $Z = \frac{\mu(x) - f^{max}}{\sigma(x)}$.

Figure 10 summarizes the Bayesian Ascent algorithm [25]. In the observation phase, the selected actions $\boldsymbol{x}^{n+1}$ are executed and the corresponding output measurement y^{n+1} is observed. The collected new data point $(\boldsymbol{x}^{n+1}, y^{n+1})$ is then used to update the regression model for the target function $f(\boldsymbol{x})$ in the learning phase of the next iteration. Additionally, depending on the improvement in the target value, the size of the trust region is adjusted to expedite the rate of convergence. If the observed increase $(y^{n+1} - f^{max})$ between the measured output y^{n+1} and the previously estimated maximum target value $f^{max}$is larger than a certain threshold (say, $\gamma(1/n)(f^{max} - y^1)$), the trust region is expanded as $\boldsymbol{\tau}^{n+1} = \beta\boldsymbol{\tau}^n$, with $\beta > 1$, in an attempt to expedite the convergence rate. Otherwise, the trust region is reset as $\boldsymbol{\tau}^{n+1} = \boldsymbol{\tau}^1$ (the size of initial trust region). In our study, we use $\gamma = 0.05$ and $\beta = 1.1$ and set the initial trust region $\boldsymbol{\tau}^1$ to be 1~10% of the input range for each input component. Imposing a trust region and adjusting its size are to ensure monotonic increase in the target value and gradual convergence to an optimum with high probability.

3.2 Cooperative Maximization of Wind Farm Power Production

This section discusses an application of the BA algorithm for finding the optimum coordinated control actions using only the measurement data observed in a physical system. In a wind farm with multiple wind turbines, wakes formed by the upstream wind turbines can decrease the attacking wind speed and, thus, the power production of the downstream wind turbines. Typically, each wind turbine would independently adjust its yaw and blades to maximize its power production without taking into consideration of the power production of other wind turbines; we refer this non-cooperative, independent actions as a greedy strategy. On the other hand, the cooperative optimization strategy would be to adjust the yaws and the blades of the wind turbines such that the

Bayesian Ascent (BA) algorithm

Choose $\boldsymbol{x}^1$ and $\boldsymbol{\tau}^1$(initial trust region size) and observe y^1

Repeat until convergence, $n = 1, 2, 3$ ...

Learning Phase:

1) Optimize the hyper parameters

$\boldsymbol{\theta}^* = \arg\max_\theta \log p\,(\boldsymbol{y}^{1:n}|\boldsymbol{x}^{1:n}, \boldsymbol{\theta})$

2) Construct a GP regression to approximate $f(\boldsymbol{x})$

$\mu(\boldsymbol{x}|\boldsymbol{D}^n) = \boldsymbol{k}^T(\mathbf{K} + \sigma_\epsilon^2\mathbf{I})^{-1}\boldsymbol{y}^{1:n}$

$\sigma^2(\boldsymbol{x}|\boldsymbol{D}^n) = k(\boldsymbol{x}, \boldsymbol{x}) - \boldsymbol{k}^T(\mathbf{K} + \sigma_\epsilon^2\mathbf{I})^{-1}\boldsymbol{k}$

Optimization Phase:

3) Select the next input by solving

$\boldsymbol{x}^{n+1} = \arg\max_{x\in \boldsymbol{A}\cap\boldsymbol{T}} \mathrm{E}[\max\{0, f(\boldsymbol{x}) - f^{max}\}|\boldsymbol{D}^n]$

where $\boldsymbol{A} := \{\boldsymbol{x}|\ \boldsymbol{x}^l \le \boldsymbol{x} \le \boldsymbol{x}^u\}$

$\boldsymbol{T} := \{\boldsymbol{x}|\ \|x_i - x_i^{max}\| < \tau_i^n \text{ for } i = 1, \dots, m\}$

Observation Phase:

4) Execute $\boldsymbol{x}^{n+1}$and observe output y^{n+1}

Append the data $\boldsymbol{D}^{n+1} = \{(\boldsymbol{x}^i, y^i)|\ i = 1, \dots, n+1\}$

5) Update the size of trust region

if $y^{n+1} - f^{max} \ge \gamma(1/n)(f^{max} - y^1)$ **then**

$\boldsymbol{\tau}^{n+1} = \beta\boldsymbol{\tau}^n$, $(\beta > 1)$

else

$\boldsymbol{\tau}^{n+1} = \boldsymbol{\tau}^1$

end if

Fig. 10. Bayesian Ascent (BA) Algorithm

total wind farm power production is maximized. The solution for the cooperative optimization problem involves the development of an analytical wind farm power function which is derived from physical principles, empirical observations and experimental calibrations [42]. Here, we employ the BA algorithm for the cooperative control optimization problem using the yaw and blade angles of the wind turbines as the input control actions $\boldsymbol{x}$ and the total power output collected from the wind turbines as the output measurement y.

A wind tunnel experiment with 6 scaled wind turbines is conducted at the KOCED's Wind Tunnel located at Chonbuk National University in Korea [43]. As shown in Figure 11, the scaled wind turbine is made of three aluminum blades with a length of 70 cm and the rotor diameter is 150 cm. The tower is made of a steel tube with a height of 100 cm. The blades are controlled by a servomotor (Dynamixel-64T) through a mechanical linkage that allows the blade pitch angles varying from $0°$ to $20°$. The yaw is controlled by the same type of servomotor through a mechanical gear system and the yaw is allowed to rotate from $-40°$ to $40°$. The ranges of the servomotors correspond to the physical constraints $\boldsymbol{A}$ imposed on the control inputs $\boldsymbol{x}^l \le \boldsymbol{x} \le \boldsymbol{x}^u$. An AC generator is used to convert the mechanical energy into electrical energy. The wind turbines are arranged as shown in Figure 12 with the longitudinal separation of 7*D* and a

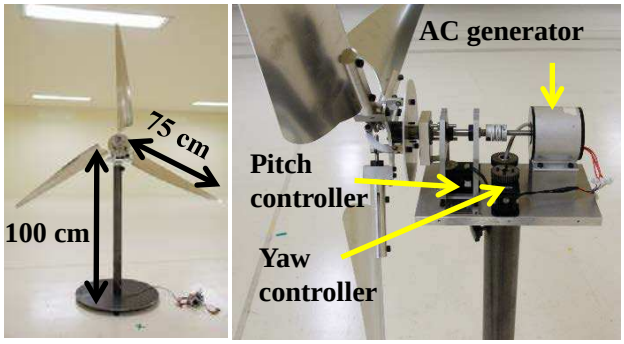


Fig. 11. Experimental design of the scaled wind turbine

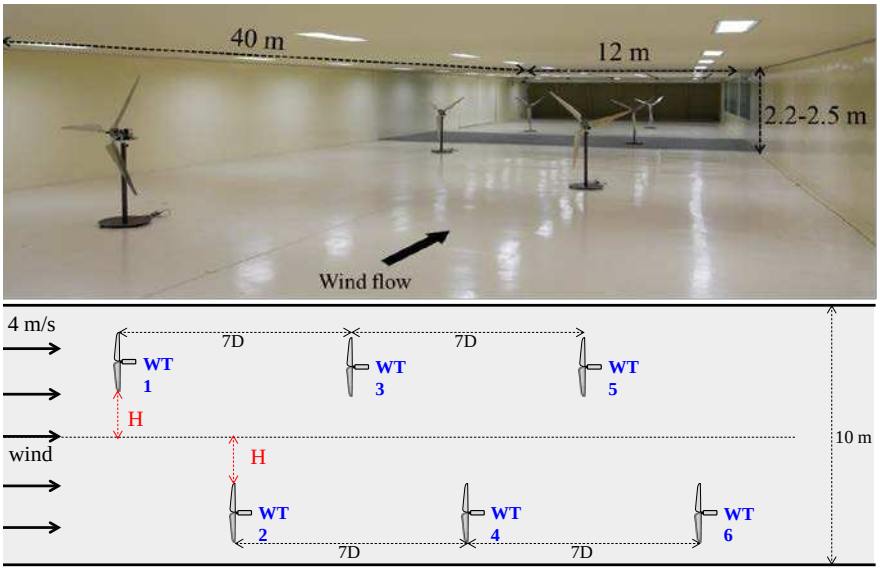


Fig. 12. Experimental set up for the 6 scaled wind turbines in a wind tunnel

lateral separation of 3m. A constant wind speed is set at 4 m/s (measured at a distance of 32 m from the front of the test section).

Three sets of experiments are conducted to assess the performance of the cooperative control of wind turbines using the BA algorithm. First, we measure the maximum freestream power P_i^F of a wind turbine i that can be produced at its location when it operates alone and without wake interference from other wind turbines. From the measurements, the absolute maximum power for all 6 wind turbines can be computed as $P^F = \sum_{i=1}^{6} P_i^F$. Second, for comparison, we measure the maximum power P_i^G of a wind turbine i that can be produced at its location when the upstream wind turbines are producing their maximum powers. For this greedy strategy, the wind farm power efficiency relative to the absolute maximum power from the freestream measurements can be computed as $\sum_{i=1}^{6} P_i^G / P^F$. Finally, the third experiment is conducted to measure the

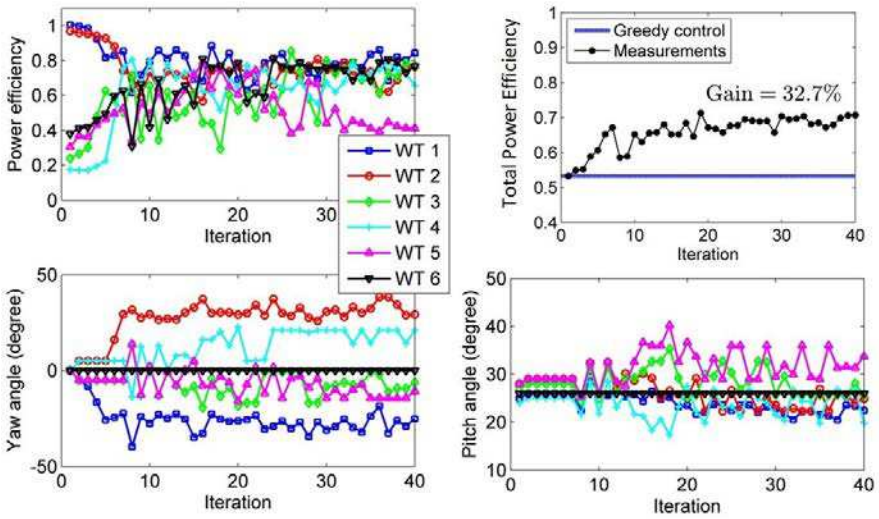


Fig. 13. Experimental results from executing BA algorithm for cooperative control

maximum power P_i^C of wind turbine i that is produced at its location when executing the BA algorithm for the cooperative strategy. Starting from the configuration obtained from the greedy control, the BA algorithm is applied to coordinate the actions of the wind turbines as an attempt to maximize the total wind farm power. The wind farm power efficiency for the cooperative control strategy is then computed as $\sum_{i=1}^{6} P_i^C / P^F$.

During the test, the blade pitch angle and the yaw offset angle of WT 6 located at the end of the wind tunnel is fixed in its position. In total, 10 input control variables, the yaw and the blade pitch (servo) angles for wind turbines WT1~5, are optimized by the BA algorithm. As shown in Figure 13, significant gain in the total power production can be observed and the maximum gain is obtained in about 20 iterations. It can be seen that the power productions by the two upstream wind turbines WT 1 and WT 2 initially are near the maximum free stream powers. During the execution of the BA algorithm, these two wind turbines offset their yaw angles the most so that the power productions of the downstream wind turbines increase, and thereby maximize the total power production increase. It is also interesting to note that WT 1 and WT 3 offset their yaw angles in clockwise direction while WT 2 and WT 4 offset in counter-clockwise direction such that the wakes are diverted away from the downstream wind turbines.

4 Summary and Discussion

Data-driven machine learning has been an active area of research and has been successfully applied in business, medical, engineering and many other domains. As engineering systems are inherently complex, dynamic, and uncertain, and the data observed in a physical environment is often noisy, probabilistic-based Bayesian learning can play an important role in developing machine learning models. The uncertainties estimated

with the learned models can provide valuable insights about the system or problem of interest and help decision making. This paper reviews two Bayesian-based approaches, namely Gaussian Process Regression and Bayesian Optimization, and demonstrates their potential applications for system characterization, diagnostic analysis and control optimization and their implementation with physical engineering systems.

The use of a non-parametric regression model, namely Gaussian Process Regression (GPR), allows the complex relationship between the input features and the target value of the system of interest to be modelled. In this paper, we discuss the use of GPR to develop predictive energy consumption model and predictive tool condition diagnostic model of a milling machine tool. In constructing predictive models useful for practical implementation, it is important to select meaningful features appropriate for the problem of interest.

- To establish the predictive power consumption model for the milling machine, we combine the information collected from a milling machine controller and the power meter. GP is then employed to model the relationship between the input machining parameters and output energy consumption and constructs a prediction function for the energy consumption with confidence bounds. It is important to note that the raw time series data are first transformed into meaningful features that are related to the numerical control (NC) code for the milling operations [32]. The GPR models can thus be used to analyze the milling operations as described with NC code and to determine optimal tool path for machining a part [13,14].
- When establishing the predictive tool condition diagnostic model, sensor (vibration and sound) data that are relevant to understand the tool conditions are collected. The time series data are pre-processed using the Welch's method that results in smoother power spectrum. Furthermore, the transformation from time series of an arbitrary length data to the frequency domain produces the data sets with specific number of points and reduces the dimension significantly. The GP model provides confidence bounds for the predictive estimations which can be useful in a practical application where the tool-condition predictions are used to determine when to change machine tools.

For both illustrative case examples, the GPR models are trained with relatively small number of experimental training data sets but the learned model is able to give reasonable predictive estimation quantified with uncertainty measures. As more data become available, the GPR models and their prediction capabilities should improve. On the other hand, one of the drawbacks of the GP, in the form that is described in this paper, is that it uses the whole set of samples or features information to perform the prediction. When the size of training data is large, approximated methods for training and prediction to reduce the computational and storage requirements may be necessary [44-47].

For control optimization, we discuss the Bayesian Ascent (BA) algorithm that optimizes a target system using limited amount of data [25]. The BA algorithm builds upon the Bayesian optimization framework but augmented with a trust region constraint. During the learning phase, the input and output relationship of a target system is modelled using Gaussian Process (GP) regression. At each iterative step, the algorithm exploits the constructed GP model function and determines the next sampling point that

can best increase the expected improvement. The trust region constraint ensures that the next input is selected from the region near the best input observed so far. Furthermore, the BA algorithm is able to increase a target value incrementally with gradual changes in the input actions. To illustrate applicability to physical problems, the BA algorithm is employed to the wind farm power maximization problem using only input (control actions) and output (wind farm power production) data. The experimental results show that the BA algorithm is able to increase the total power production by gradually changing the control actions of the wind turbines. Without explicitly constructing the objective function, the BA algorithm is able to optimize the operations of a complex physical system using only the measurement data from the system.

While this paper focuses the discussion on the Gaussian Process Regression and Bayesian Optimization, there exist a broad range of machine learning techniques. It should be noted that selection of an appropriate machine learning technique is problem and data dependent and would require judicious engineering knowledge of the problem involved. As computing and sensing technologies advance, the scope of machine learning tasks will continue to grow. There is no doubt that we will continue to see the impacts of data-driven machine learning in engineering for years to come.

Acknowledgments and Disclaimer

The authors would like to acknowledge the assistance from the late Prof. David Dornfeld of the Laboratory for Manufacturing and Sustainability at UC Berkeley and Dr. Raunak Bhinge of Infinite Uptime, Inc., who conducted the experiments and collected the machine operation data on the Mori Seiki Milling Machine. The authors would also like to thank Prof. Soon-Duck Kwon of Chonbuk National University in Korea, who generously provided the wind tunnel laboratory (KOCED Wind Tunnel Center) facilities for the wind farm power production experiments.

The work described in this paper was partially supported by the National Institute of Standards and Technology (NIST) cooperative agreement with Stanford University (Grant No. 70NANB12H273 and 70NANB17H031), and the US National Science Foundation (NSF) (Grant No. ECCS-1446330). Any opinions, findings, conclusions or recommendations expressed in the paper are solely those of the authors and do not necessary reflect the views of NSF, NIST and the authors' collaborators. Certain commercial systems are identified in this paper; such identification does not imply recommendation or endorsement by NSF, NIST, Stanford University or the authors; nor does it imply that the products identified are necessarily the best available for the purpose.

References

1. Jordan, M. I., Mitchell, T. M: Machine learning: trends, perspectives and prospects. Science 349, 255-269 (2015).
2. Wuest, T., Weimer, D., Irgens, C., Thoben, K-D.: Machine learning in manufacturing: advantages, challenges, and applications. Production & Manufacturing Research: An Open Access Journal 4, 23-45 (2016).

3. Suresh, P. V. S., Venkateswara Rao, P., Deshmukh, S. G.: A genetic algorithmic approach for optimization of surface roughness prediction model. International Journal of Machine Tools and Manufacture 42, 675-680 (2002).
4. Ghosh, N., Ravi, Y. B., Patra, A., Mukhopadhyay, S., Paul, S., Mohanty, A. R., Chattopadhyay, A. B.: Estimation of tool wear during CNC milling using neural network-based sensor fusion. Mechanical Systems and Signal Processing 21, 466-479 (2007).
5. Ak, R., Helu, M., Rachuri, S.: Ensemble neural network model for predicting the energy consumption of a milling machine. In: 2015 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference (IDETC/CIE 2015), ASME, pp. V004T05A056 (2015).
6. Jihong, Y., Lee J.: Degradation assessment and fault modes classification using logistic regression. Journal of Manufacturing Science and Engineering 127, 912-914 (2005).
7. Carbonneau, R., Laframboise, K., Vahidov, R.: Application of machine learning techniques for supply chain demand forecasting. European Journal of Operational Research 184, 1140-1154 (2008).
8. Ghahramani, Z.: Probabilistic machine learning and artificial intelligence. Nature 521, 452-459 (2015).
9. Orbanz, P., Teh, Y. W.: Bayesian nonparametric models. In: Encyclopedia of Machine Learning, Springer, USA, pp. 81-89 (2011).
10. Murphy, K. P.: Machine learning: a probabilistic perspective. MIT Press, Cambridge, MA (2012).
11. Rasmussen, C. E.: Gaussian processes in machine learning. In: Advanced Lectures on Machine Learning, Springer Berlin Heidelberg, New York, pp. 63–71 (2004).
12. Rasmussen, C. E., Williams, C. K. I.: Gaussian processes for machine learning, MIT Press, Cambridge, MA (2006).
13. Bhinge, R., Park, J., Law, K.H., Dornfeld, D., Moneer, M., Rachuri, S.: Towards a generalized energy prediction model for machine tools. Journal of Manufacturing Science and Engineering 139(4), 041013 (2017).
14. Park, J., Law, K. H., Bhinge, R., Biswas, N., Srinivasan, A., Dornfeld, D., Helu, M., Rachuri, S.: A generalized data-driven energy prediction model with uncertainty for a milling machine tool using Gaussian Process. In: 2015 International Manufacturing Science and Engineering Conference (MSEC2015), ASME, pp. V002T05A010 (2015).
15. Nguyen-Tuong, D., Peters, J. R., Seeger, M.: Local Gaussian process regression for real time online model learning. In: 22nd Annual Conference on Neural Information Processing Systems, Advances in Neural Information Processing Systems 21, pp. 1193–1200 (2008).
16. Teramura, K., Hideharu, O., Yuusaku, T., Shimpei, M., Shinichi, M.: Gaussian process regression for rendering music performance. In: International Conference on Music Perception and Cognition (ICMPC 10), pp. 167–172 (2008).
17. Alpaydm, E.: Introduction to machine learning, 3rd edition, MIT Press (2014).
18. Brochu, E., Cora, M. V., Freitas, N.: A Tutorial on Bayesian Optimization of Expensive Cost Functions with Application to Active User Modelling and Hierarchical Reinforcement Learning. Tech. Rep. arXiv:1012.2599, University of British Columbia, Canada (2010).
19. Jones, D., Schonlau, M., Welch, W.: Efficient global optimization of expensive black-box functions. Journal of Global Optimization 13, 455-492 (1998).
20. Osborne, M.: Bayesian Gaussian Process for Sequential Prediction, Optimization and Quadrature. PhD Dissertation, Department of Computer Science, University of Oxford, UK (2010).
21. Bubeck, S., Munos, R., Stoltz, G., Szepesvari. C.: X-armed Bandits. Journal of Machine Learning Research 12, 1655-1695 (2011).

22. Scott, S.L.: A modern Bayesian look at the multi-armed bandits. Applied Stochastic Models in Business and Industry, 26(6), 639-658 (2010).
23. Osborne, M., Garnett, R., Roberts, S.: Active data selection for sensor networks with faults and changepoints. In: IEEE International Conference for Advanced Information Networking and Applications, DOI: 10.1109/AINA.2010.36 (2010).
24. Garnett, R., Osborne, M., Roberts, S.: Bayesian optimization for sensor set selection. In: Proceedings of the 9th ACM/IEEE International Conference on Information Processing in Sensor Network, pages 209-219 (2010).
25. Park, J., Law, K. H.: Bayesian Ascent: A Data-Driven Optimization Scheme for Real-Time Control with Application to Wind Farm Power Maximization. IEEE Transactions on Control Systems Technology 24(5), 1655-1668 (2016).
26. Duvenand, D.K.: Automatic Model Construction with Gaussian Processes. PhD Thesis, University of Cambridge (2014).
27. Genton, M.G., Classes of kernels for machine learning: a statistics perspectives. Journal of Machine Learning Research 2, 299-312 (2001).
28. Rasmussen, C. E., Nickisch, H.: Gaussian processes for machine learning (GPML) toolbox. Journal of Machine Learning Research 11:3011–3015 (2010).
29. Blondel, M., et al.: Scikit-learn, machine learning in Python. http://scikit-learn.org/stable/, last viewed 2016/12/22.
30. Sobel, W.: MTConnect Standard. Part 1 — Overview and Protocol. Version 1.3.0 (2015).
31. Vijayaraghavan, A., Dornfeld, D.: Automated energy monitoring of machine tools. CIRP Annals – Manufacturing Technology 59, 21-24 (2010).
32. Bhinge, R., Biswas, N., Dornfeld, D., Park, J., Law, K. H., Helu, M., Rachuri, S.: An intelligent machine monitoring system for energy prediction using a Gaussian process regression. In: IEEE International Conference on Big Data, pp. 978–986, IEEE (2014).
33. Fan, J., Han, F., Liu, H.: Challenges of big data analysis. National Science Review 1(2) 293–314 (2014).
34. Kannatey-Asibu, E., Dornfeld, D.A.: A study of tool wear using statistical analysis of metal-cutting acoustic emission. Wear 76(2), 247–261 (1982).
35. Dimla, D., Lister, M.: Online metal cutting tool condition monitoring - I: force and vibration analyses. International Journal of Machine Tools and Manufacture 40(5), 739–768 (2000).
36. Stoica, P., Moses, R. L.: Spectral Analysis of Signals, Vol. 452. Pearson Prentice Hall Upper Saddle River, NJ (2005).
37. Allen, R. L., Mills, D.: Signal Analysis: Time, Frequency, Scale, and Structure. John Wiley & Sons (2004).
38. Welch, P.: The use of Fast Fourier Transform for the estimation of power spectra: a method based on time averaging over short, modified periodograms. IEEE Transactions on Audio and Electroacoustics 15(2), 70–73 (1967).
39. Williams, C. K., Rasmussen, C. E.: Gaussian processes for regression. Advances in Neural Information Processing Systems, pp. 514–520 (1996).
40. Wilson, A. G., Adams, R. P.: Gaussian process kernels for pattern discovery and extrapolation. In: ICML'13 Proceedings of the 30th International Conference on International Conference on Machine Learning, 28: III-1067-III-1075 (2013).
41. Mockus. J., Fretitas, A., Castelanous, J.A.: Toward Global Optimiza-tion, North-Holland, Amsterdam (1978).
42. Park, J., Law, K.H.: Cooperative wind turbine control for maximizing wind farm power using sequential convex programming. Energy Conversion and Management 101, 295-316 (2015).

43. Park, J., Kwon, S., Law, K.H.: A data-driven, cooperative approach for wind farm control: a wind tunnel experimentation. Energies 10, 852 (2017).
44. Snelson, E., Ghahramani, Z.: Sparse Gaussian processes using pseudo-inputs," In: Neural Information Processing Systems (NIPS) Conference, pp. 1257–1264 (2005).
45. Quiñonero-Candela, J., Rasmussen C.E.: A unifying view of sparse approximate Gaussian process regression. Journal of Machine Learning Research 6, 1939–1959 (2005).
46. Ranganathan, A., Yang, M. H., Ho, J.: Online sparse Gaussian process regression and its applications. IEEE Transaction on Image Processing 20(2), 391–404 (2011).
47. Park, J., Bhinge, R., Law, K.H., Dornfeld, D., Mason, C., Rachuri, S.: Real time energy prediction for a milling machine tool using sparse Gaussian process regression. In: International Conference on Big Data, pp. 1451-1460 (2015).