Journal of Intelligent Manufacturing
https://doi.org/10.1007/s10845-022-01999-w

Stability modeling for chatter avoidance in self-aware machining: an application of physics-guided machine learning

Noel P. Greis[1] · Monica L. Nogueira[1] · Sambit Bhattacharya[2] · Catherine Spooner[2] · Tony Schmitz[3]

Received: 1 January 2022 / Accepted: 16 July 2022

Abstract

Physics-guided machine learning (PGML) offers a new approach to stability modeling during machining that leverages experimental data generated during the machining process while incorporating decades of theoretical process modeling efforts. This approach addresses specific limitations of machine learning models and physics-based models individually. Data-driven machine learning models are typically black box models that do not provide deep insight into the underlying physics and do not reflect physical constraints for the modeled system, sometimes yielding solutions that violate physical laws or operational constraints. In addition, acquiring the large amounts of manufacturing data needed for machine learning modeling can be costly. On the other hand, many physical processes are not completely understood by domain experts and have a high degree of uncertainty. Physics-based models must make simplifying assumptions that can compromise prediction accuracy. This research explores whether data generated by an uncertain physics-based milling stability model that is used to train a physics-guided machine learning stability model, and then updated with measured data, domain knowledge, and theory-based knowledge provides a useful approximation to the unknown true stability model for a specific set of factory operating conditions. Four novel strategies for updating the machine learning model with experimental data are explored. These updating strategies differ in their assumptions about and implementation of the type of physics-based knowledge included in the PGML model. Using a simulation experiment, these strategies achieve useful approximations of the underlying true stability model while reducing the number of experimental measurements required for model update.

Keywords Physics-guided machine learning · Informed machine learning · Stability modeling · Milling · Machine learning

Introduction

In the field of high-speed machining with processes like milling, turning, and drilling, simultaneous achievement of the performance targets of part accuracy, high surface finish and productivity is often constrained by the occurrence of dynamic instability of the machining process, referred to as chatter. Improvements in process control due to new machining technologies have typically improved one or more of these metrics, but often at the expense of deterioration in the other(s) reflecting a trade-off between process control and productivity. Over the last 50 years, many new technologies such as computer numerically controlled (CNC) machining have allowed greater process control while increasing the capability and flexibility of the process itself. Computer-aided design/computer-aided manufacturing (CAD/CAM) software has enabled digital representation of

✉ Noel P. Greis
ngreis@uncc.edu; noelgreis@gmail.com

Monica L. Nogueira
mnogueir@uncc.edu

Sambit Bhattacharya
sbhattac@uncfsu.edu

Catherine Spooner
casabelle42@gmail.com

Tony Schmitz
tony.schmitz@utk.edu

[1] University of North Carolina at Charlotte, Charlotte, USA

[2] Fayetteville State University, Fayetteville, USA

[3] University of Tennessee, Knoxville 5, USA

part geometries and the subsequent generation of computer instructions to command the machine motions required to remove material and leave the desired part geometry. While these technologies have increased the spindle speed range due to new spindle designs, as well as improved tool materials and coatings, the machining parameters are still typically selected based on manufacturer recommendations or, in most cases, operator experience. In practice, operators typically adjust spindle speeds downwards when encountering chatter, reducing productivity. However, in many cases increasing spindle speeds can return stability to the process and improve productivity. Knowledge of the true, as opposed to physics-based or manufacturer-provided, stability model would enable operators to better select initial machining parameters to both maintain dynamic stability and improve part quality without compromising productivity.

Knowledge of the true operational stability model would also enable operators—and ultimately the machines themselves—to make in-process optimum parameter adjustments during production. As machines become more intelligent, self-knowledge of the true stability model is the foundation for self-aware operations. Self-aware machines, as their name suggests, exhibit self-awareness of both their operational health and machining status and, as required, self-control to perform parametric adjustments that maintain continued performance to target levels. For example, self-aware machines can make parametric adjustments to keep themselves operational while waiting for service; can adjust their machining parameters to assure process stability during machining; and can adjust their load to balance production yields in their cell in the event of excess demand or machine downtime. This level of machine intelligence has considerable potential to enhance productivity in the manufacturing environment and to maintain optimal operational performance for maximum efficiency. New sensor technologies and machine learning (ML) methods are building the foundations for self-awareness that would enable the machining process to identify those combinations of spindle speed and axial depth-of-cut needed to maintain dynamic stability of the process and, in the case of approaching chatter, identify the adjustments in these parameters that allow the process to quickly return to stability.

This research builds on a recent trajectory of research in machine learning referred to as physics-informed or physics-guided machine learning (PGML). To implement the PGML approach, a machine learning model is trained with stability data simulated using a physics-based model built on the well-known receptance coupling substructure analysis (RCSA), a mechanistic force model, and Fourier analysis to generate the stability lobe diagram (SLD). The SLD defines regions of stability and instability as a function of spindle speed and depth of cut, described in more detail later. This pre-trained model is then updated with experimental measured data, along with domain knowledge, to enable the convergence of the physics-based model to the true operational model. Physics-based theory also informs the updating strategies by guiding the sampling of measurement points prior to retraining the PGML. A major contribution of this paper is the exploration of measurement strategies that increase predictive accuracy of the PGML model while minimizing measurement cost. Experiments are implemented within a simulation framework that provides a common benchmark for evaluating the updating strategies. The research goal is to develop guidelines for sampling that promote fast and inexpensive convergence of the physics-based model to the true operational stability model suitable for use within a production environment.

This paper proceeds as follows. Sect. 2 introduces PGML and reviews recent literature in this area. Sect. 3 provides a perspective on the challenge of maintaining dynamic stability during machining motivating this work. Sect. 4 describes the development of the physics-based dynamic stability model and discusses model uncertainties that motivate the PGML approach. Sect. 5 introduces the PGML approach implemented in this research. Sect. 6 presents the research framework including machine learning methods and measurement updating strategies. Sect. 7 describes the methods for both training and updating the PGML model with simulated measurement data, domain knowledge and physical theory. Numerical results are presented in Sect. 8. Implications of this research for implementation in a production environment and as a step towards self-aware machining are discussed in Sect. 9 followed by conclusions in Sect. 10.

Physics-guided machine learning

When modeling complex machining processes, the practical choice—and associated best practice—has been to choose between physics-based or data-driven models for prediction. Both approaches have distinct advantages when applied to complex systems with integrated mechanical, electrical, and software components. However, individually, they fall short of delivering the broad capability and accuracy needed for chatter-free high precision machining. Informed machine learning is part of an emerging trend to embed domain knowledge, broadly defined, into ML models. This approach use physical principles to inform and guide the search for the best data-driven model, thereby capturing the best attributes of both physics-based and data-learning models, as shown in Fig. 1 (Greis et al., 2020).

Given the diverse scientific and other disciplines in which these techniques are being applied, this approach has been referred to by several names such as physics-guided

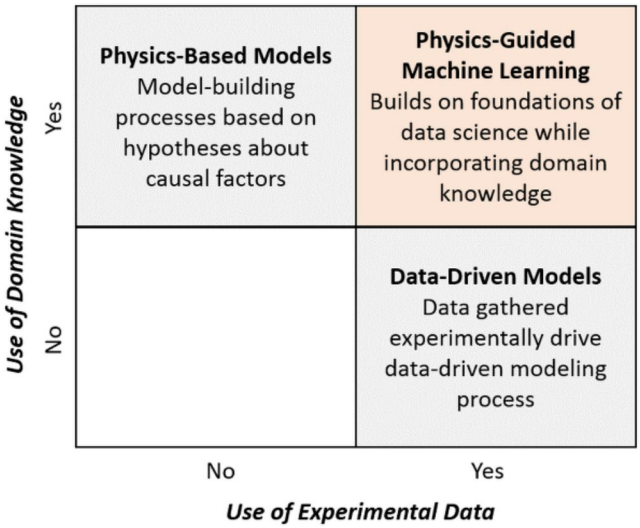

Fig. 1 Physics-Guided Data Learning

ML, physics-informed ML, physics-based ML, physics-aware ML—or simply informed ML. A number of review papers have appeared in the last several years that offer various frameworks for organizing the theory and practice of informed machine learning (Kim et al., 2021). Willard et al. (2020) offer a taxonomy of physics-based modeling with machine learning techniques by application area and class of methodology. von Rueden et al. (2021) offer a taxonomy that considers the source of the knowledge (i.e., natural science, expert knowledge, or accepted physical law), its representation (i.e., algebraic equations, simulation results, or logical rules), and how it is integrated into the machine learning pipeline. Roscher et al. (2020) approach informed machine learning from the perspective of its ability to extract and explain novel scientific results. More recently, Karniadakis et al. (2021) offer a broad review of prevailing trends in informed machine learning, including capabilities and limitations. In addition, a growing body of research on physics informed neural networks (PINNs) focus on integration of physical knowledge that can be described as partial derivative equations directly in the neural network structures or their loss functions (Cuomo et al., 2022; Raissi et al., 2017a, b, 2019). As is evident from the current literature, a definitive nomenclature has not yet emerged. This research implements three approaches to incorporate physics-based knowledge into the model. A physics-based stability model generates the simulation data that determine the baseline machine learning model. Domain knowledge that can be considered to be expert knowledge captured in logical rules informs the determination of non-measured stability value during updating. And mathematical expressions of the stability model guide the measurement sampling process during updating. Given these different approaches, the overall method proposed herein is referred as PGML, recognizing that the naming choice is not straightforward given research streams in the current literature.

Data-driven approaches are built on large sets of historical data and can learn directly from real-time sensor data (e.g. vibration, temperature, acoustic emissions, etc.) collected during machining. Advantages include the ability to model highly complex physical systems for which there is no underlying physical model that completely defines the system, or where the relationships between the input and output variables are difficult to describe using simple mathematics, or when the ability to include contextual data (e.g. environmental conditions, changes in operating regime, etc.) is important. A challenge with black box data-driven models is that they are agnostic to physical law because they rely only on data and not theory. They are, also, therefore dependent on the available data for training which can lead to relationships that do not generalize beyond the range of the training dataset. Since data-driven model predictions are generally limited to what the training set has *seen*, these models are not always useful for generating new scientific knowledge. Physics-based models are still preferred for scientific discovery. However, PGML models have been shown to be capable of revealing new knowledge. These insights can be attributed in part to the inclusion of domain knowledge of the underlying physics explicitly into the PGML modeling process.

The application of machine learning to stability prediction in milling, in particular, has been an area of keen and accelerating interest (Sharp et al., 2018; Oleaga et al., 2018; Cherukuri et al., 2019; Tao et al., 2018) and several excellent review papers have been published (Kim et al., 2018; Wang et al., 2018). In addition, Karandikar et al. (2020) describe a novel Bayesian learning approach for stability boundary and optimal parameter identification in milling without the knowledge of the underlying tool dynamics or material cutting force coefficients using a limited number of data points. Cornelius et al. (2021) introduce a Bayesian framework for identifying the milling stability boundary and system parameters for reverse parameter identification through iterative testing. Several recent papers incorporate data collected during simulation and real-time operations to the creation and adaptation of stability models. Friedrich et al. (2017; 2018) implement machine learning to predict chatter using scenario-specific milling data obtained from a simulation model, and then extend that model to allow for continuous learning and time-variant systems using measured acceleration signals. Saadallah et al. (2018) implement a machine learning framework based on results of a geometric physically-based simulation with varied process parameter values and refined using an active learning approach.

PGML offers the next advance by formally incorporating physical theory and domain knowledge of the underlying physics into the machine learning pipeline. Early work in PGML focused on non-manufacturing applications characterized by degrees of complexity that resist capture by traditional physical models, but for which large amounts of data are available for machine learning (Karpatne et al., 2017). These include physical phenomena such as turbulent flow (Singh et al., 2017) and geoscience applications such as hydrologic modeling and climate change (Karpatne et al., 2017; Sheikh & Jahirabadkar, 2018; Faghmous & Kumar, 2014). Areas of application have expanded recently into other domains including structural dynamics (Yu et al., 2020), power system management (Wang et al., 2020a), and modeling of seismic events (Zhang et al., 2020).

PGML is a relatively newly explored methodology in machining. Several recent applications can be noted. Wang et al. (2020b) construct a physics-guided gated recurrent units (GRU) machine learning model for continuous prediction of tool wear that considers complex tool cutting conditions and dynamic changes of physical parameters experienced in practice. Lee et al., (2018) introduce a physics-based artificial neural network for online monitoring of steady-state tool temperatures at the tool/chip interface. Lu et al. (2017) model the electrochemical micro-machining process by embedding knowledge about the relationship between the input process parameters and intermediate outputs, as well as domain knowledge about the mechanisms of the micro-machining process, into a neural network structure.

Most recently, physics-informed or physic-guided approaches for stability modeling during machining operations have been implemented within a transfer learning environment that allows for the effects of the changing process dynamics and highly complex cutting operations. Postel et al. (2020) utilize deep neural networks pre-trained with simulated data to match network predictions with experimentally observed stability states acquired under different cutting conditions using ensemble transfer learning to combine predictions. Similarly, Unver & Sener (2021) combine analytical solutions and convolutional neural networks within a transfer learning framework. The primary classifier AlexNet is pre-trained on analytically developed and labeled stability solutions before being fed experimental vibration data collected during milling. Yesilli et al. (2020) use wavelet packet transform and ensemble empirical mode decomposition, two well-known chatter detection algorithms, to not only classify acceleration signals associated with chatter, but also to transfer knowledge from one cutting configuration to another.

Challenges of dynamic stability modeling for chatter detection

Dynamic stability of cutting processes such as milling is a function of the dynamic behavior of the tool and the workpiece during cutting. CAD/CAM software generally treats machining as a geometric effort. As long as the cylindrical tool follows the required path through the prismatic work material imparting the desired geometry, it is assumed that the machining process is acceptable. This approach does not consider constraints imposed by machining dynamics. For example, some combinations of spindle speed and axial depth-of-cut will exhibit self-excited vibrations, or chatter, which produce large forces, vibrations, unacceptable surface finish, and potential damage to the tool, part, and spindle. Additionally, even if stable behavior is obtained, the geometric accuracy of the machined part may or may not satisfy design tolerances, again depending on the selected spindle speed and axial depth-of-cut combination. Stability models that capture the true machining dynamics on the shop floor are desired to select spindle speed and axial depth-of-cut combinations that avoid chatter, while providing the required geometric accuracy.

Combinations of spindle speed and axial depth-of-cut that result in a stable machining process are represented by a stability lobe diagram (SLD) which defines stable and unstable stability regimes as a function of these two machining parameters. SLDs can be computed analytically or experimentally. In order to determine the SLD experimentally, a large number of combinations of spindle speed and axial depth-of-cut must be tested for chatter which is time-intensive, as well as materially and computationally expensive. Analytical and physics-based models rely on theory derived through experiment and expressed notationally. A limitation of stability models and SLDs derived analytically is that they rarely exactly mirror the SLDs that govern operations on the factory floor. The combinations of spindle speed and axial depth-of-cut that yield stable dynamics in an operational environment depend not only on model uncertainties, but also the dynamics of the particular milling machine, conditions in the ambient operational environment, and also conditions such as tool wear.

Further, an SLD calculated once is only true for that machine working on that workpiece and in that environment, and only for that time and not for the whole lifetime of that machine. To analyze the underlying, and operational, stability model, measurements must be repeated periodically as operating or machine conditions change over time. While a manufacturer may provide operational guidelines for initial parameter setting, they are not specific to the particular production environment, the age of the machine, or the acquired skill of the particular operator. Thus, analytical

models fail to meet the essential criterion for self-awareness—a stability model and associated SLD that have been calibrated to the production environment and that can provide feedback to an operator for manual adjustments in real time to regain stability at the onset of chatter and ultimately to the machine itself for automatic self-adjustment.

The lack of alignment of the physics-based model and true SLD due to its time-varying nature present both computational and operational challenges for chatter detection in a production environment. For setting initial cutting parameters, a method is required that can align the physics-based or manufacturer-supplied SLD with the true SLD so the operator can select initial parameters. As the machine performs over time, an automated method is required to update the physics-based SLD as conditions change. Thus, the model must be capable of handling time-varying conditions. The method also has to efficiently accommodate continuously collected training data since all the measured data may not be collected by experiment at one time, but periodically during production. As manufacturers move toward self-aware machining, this data may be automatically collected, chatter detected, and the SLD adjusted autonomously, so that the machine itself is able to self-adjust cutting parameters to maintain stability.

Development of the physics-based dynamic stability model

The PGML approach implemented here begins with the development of a physics-based model for milling. Milling is a machining process that uses a rotating cylindrical cutting tool to remove material from the surface of a workpiece. During the milling process, the cylindrical tool follows a predetermined tool path to achieve the desired geometry of the workpiece. As discussed above, under certain dynamic conditions, the milling process will exhibit chatter—a self-excited vibrational state that leads to instability and uncontrollability of the system. Development of physics-based models to predict chatter during milling are based on an understanding of the vibrational behavior of the tool-holder-spindle-machine assembly (and sometimes the part) which imposes specific measurement requirements for model-building. First, the vibrational behavior at the tool tip is traditionally described by the frequency response function (FRF) which is obtained through modal testing. A popular approach is to excite the structure in question using an instrumented hammer and to use a low mass accelerometer attached to the structure to record the subsequent time-domain vibration response. The frequency domain displacement-to-force ratio is the FRF, or receptance, for the tool tip (or part).

While the measurement procedure is well-understood, the lack of widespread availability of modal testing equipment and associated expertise has hindered the implementation of machining modeling on the shop floor. This points to a second need; the tool tip receptances must be identified without physical measurement of each. Because the tool is clamped in a holder that is inserted in a spindle attached to the machine, tool tip receptance prediction is not trivial. Additionally, the tool tip receptance and machining models are deterministic, but include inherent uncertainties. The predicted machining parameters are, therefore, also uncertain. This establishes the need for uncertainty reduction through testing. To build the physics-based dynamic stability model and its SLD, three physics-based models are used. First, receptance coupling substructure analysis (RCSA) is used to predict the tool tip receptance. Second, a mechanistic force model is used to relate the cutting force to the commanded chip area through cutting force coefficients. Third, a mean force frequency domain analysis is used to predict the stability limit using the first two models as input. These models are briefly described in the following sections.

Receptance coupling substructure modeling

Using three-component RCSA for tool tip dynamics prediction has been previously demonstrated (Schmitz & Donaldson, 2000; Schmitz & Duncan, 2005; Schmitz & Smith, 2009). In prior efforts, the free-free boundary condition tool and holder were modeled as cylindrical cross-section Timoshenko beams. These beam receptances were coupled analytically to measured receptances of the spindle-machine. The sequence of steps for tool point receptance prediction are: (1) calculate the tool receptances (free-free boundary conditions) using the Timoshenko beam model; (2) calculate the holder receptances (free-free boundary conditions) using the Timoshenko beam model; (3) measure the spindle-machine receptances using impact testing; and (4) couple these receptances to predict the tool-holder-spindle-machine assembly dynamics using either rigid or flexible-damped compatibility conditions. The approach is summarized in Fig. 2.

The coupling procedure is described by Fig. 3 and Eqs. (1)-(3) below. Figure 3 shows that both linear and translational coordinates are required. This leads to four receptances that relate displacement, x, and rotation, θ, to force, f, and moment, m.

The four direct receptances are shown in Eq. 1 for the right end of the cylinder (tool) pictured in Fig. 3. The four component receptances are organized into a generalized coordinate format in Eq. 2. Using this formulation, the direct tool point receptances (uppercase coordinates) are computed for a rigid connection using the component direct and

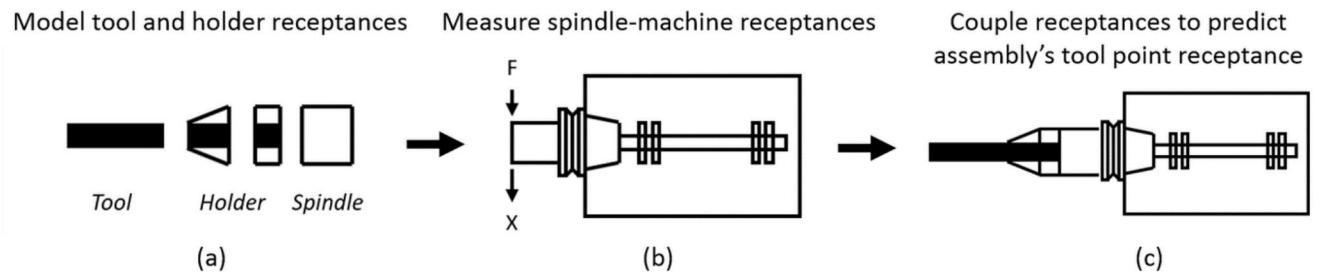

Fig. 2 Summary of RCSA Procedure

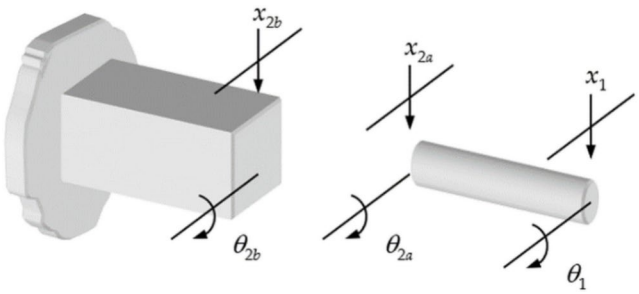

Fig. 3 RCSA Coordinates

cross receptances (lowercase coordinates) in Eq. 3. Flexible/damped connections may also be selected. In Eq. 3, the displacement-to-force H_{11} receptance provides the input to machining simulations.

$$\begin{array}{ll} h_{11} = \frac{x_1}{f_1} & l_{11} = \frac{x_1}{m_1} \\ n_{11} = \frac{\theta_1}{f_1} & p_{11} = \frac{\theta_1}{m_1} \end{array} \tag{1}$$

$$\left\{ \begin{array}{c} x_1 \\ \theta_1 \end{array} \right\} = \left[\begin{array}{cc} h_{11} & l_{11} \\ n_{11} & p_{11} \end{array} \right] \left\{ \begin{array}{c} f_1 \\ m_1 \end{array} \right\}, \{u_1\} = [R_{11}]\{q_1\} \tag{2}$$

$$G_{11} = R_{11} - R_{12a}(R_{2a2a} + R_{2b2b})^{-1} R_{2a1} = \left[\begin{array}{cc} H_{11} & L_{11} \\ N_{11} & P_{11} \end{array} \right] \tag{3}$$

Force model

In mechanistic force modeling for milling, the cutting force coefficients, k, are calculated using the commanded axial depth-of-cut, b, and chip thickness, h. The model described in Eq. 4 includes force components that are tangential, t, and normal, n, to the rotating cutting edge. Force coefficients that relate the force to chip area, bh, are identified by a c subscript (cutting or shearing force). Those that relate the force to axial depth alone have an e subscript (edge or rubbing force).

$$\begin{array}{l} F_t = k_{tc} bh + k_{te} b \\ F_n = k_{nc} bh + k_{ne} b \end{array} \tag{4}$$

These coefficients may be determined by experiment where the cutting force is measured using a force dynamometer and the commanded axial depth-of-cut and chip thickness are known. Linear regression over a range of chip thickness values and nonlinear least squares fitting to the time-domain force have been applied (Rubeo & Schmitz, 2016). As an alternative, the material behavior can be defined using a constitutive model and the cutting force predicted using finite element simulation (Shi & Liu, 2004).

Frequency domain stability analysis

The analytical stability limit may then be determined using Fourier force analysis to transform the dynamic milling equations into a time-invariant, but radial, immersion dependent system (Altintas & Budak, 1995). This analysis expands the frequency domain dynamic milling equations into a Fourier series. The series is then truncated to include only the mean component. The tool point receptance and force model are inputs to the analysis. The dynamic milling process model is derived by considering the Fourier series expansion of the time-varying milling force coefficients. The eigenvalues of the dynamic milling expression are calculated analytically by selecting a chatter frequency around the dominant structural modes. Noting that the axial depth-of-cut is always a real quantity, the chatter free axial depths-of-cut and spindle speeds are analytically formulated as a function of the tool tip receptances, the cutting force coefficients, the number of teeth, milling orientation (up and down), and radial depth-of-cut.

The individual stability lobes that determine the SLD are computed by, first, selecting a chatter frequency near a dominant mode from the tool tip receptances and solving the eigenvalue equation. The critical axial depth-of-cut and spindle speed are calculated for each stability lobe. This procedure is repeated by scanning the chatter frequencies around all dominant modes from the tool tip receptance. The final output is the limiting axial depth-of-cut as a function of spindle speed for a selected radial depth-of-cut, milling orientation, and number of teeth on the endmill. The graphical

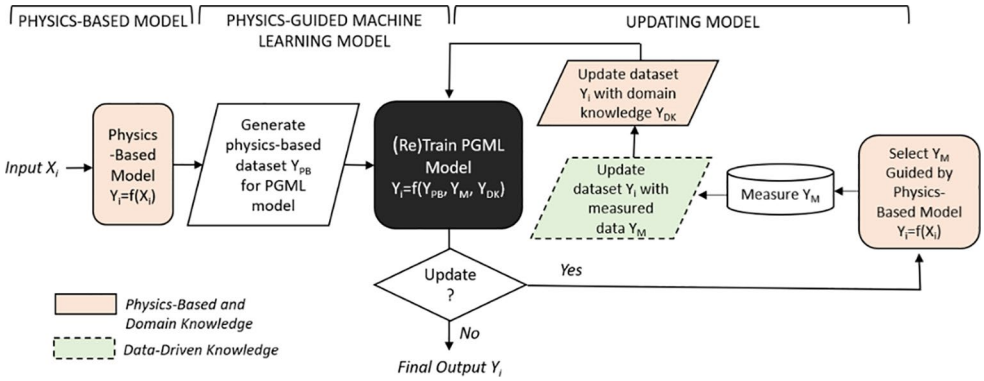

Fig. 4 Physics-Guided Machine Learning Approach

representation of this output is the SLD that defines the stability boundary that is the subject of this research.

In addition to the operational and model uncertainties noted earlier, the deterministic models described in the previous section also include measurement uncertainty. For example, the actual extension length of the endmill from the holder is subject to set-up and measurement uncertainties. This results in uncertainty in the tool tip receptance which, in turn, leads to uncertainty in the stability limit. Propagation of uncertainties in the tool and holder models, spindle receptances, and cutting force coefficients to uncertainty in the stability limit may be computed using Monte Carlo simulation (Karandikar et al. 2010). This provides a predictive model, where a probabilistic, rather than deterministic, stability limit is presented. However, if a test is performed to determine the actual stability behavior of a spindle speed-axial depth combination, there is no straightforward mapping between this result and input parameters.

PGML approach to stability modeling

The PGML approach developed herein is described conceptually in Fig. 4. The approach combines two models. First, a PGML model is created from the physics-based dynamic stability model described in the previous section. The physics-based stability model is used to generate an initial training dataset that is comprised of three parameters, the target stability state—unstable (or chatter) and stable (or no chatter)—and two physics-based input parameters spindle speed and axial depth-of-cut. A machine learning method then "fits" a baseline PGML model to this initial physics-based training dataset. The baseline PGML model should have good fit with the physics-based data to serve as a benchmark for determining predictive accuracy improvements during updating with measured data and domain knowledge. Then, in an iterative process, points in the initial physics-based training dataset are updated with measured (off-line or in-process) stability values; points whose true stability state is known by domain knowledge are also updated. The PGML model is retrained at each iteration. The points to be sampled for measurement at each iteration are selected according to novel updating strategies. As the updating process proceeds iteratively, the PGML SLD is hypothesized to converge toward the SLD of the true stability model. As convergence occurs, the predictive accuracy of the PGML model increases proportionately.

Physics-based knowledge is incorporated by three mechanisms shown by shaded red boxes in Fig. 4 with the solid outline. First, the physics-based stability model on the left generates an initial physics-based training dataset composed of physics-based stability values for combinations of spindle speed and axial depth-of-cut. The baseline PGML stability model is trained using this baseline training dataset. Second, selection of points to be measured experimentally or on the production floor is guided by the physics-based SLD. Third, non-measured data points that are known to have true stability (stable or unstable) are updated by domain knowledge. Consistent with the framework in Fig. 4, numerical simulation experiments were performed to test whether a physics-based stability model, with associated operational, model and measurement uncertainty, can be updated with measured data and domain knowledge to better approximate the true underlying stability model and, thus, improve predictions of instability.

Since the true underlying stability model and its SLD are typically unknown in practice, both physics-based and true stability models are simulated for these experiments. For the physics-based model described in Sect. 4, errors are intentionally introduced into the model inputs so that the predicted stability limit used to train the PGML model includes uncertainty. However, because the input errors are known, it is possible to determine the true (zero uncertainty) stability limit as well. This error-free stability model is used to generate true experimental data, which replaces the uncertain stability limit defined in the uncertain initial training dataset during updating.

For the purposes of these numerical experiments, the physics-based SLD is simulated as follows. Errors are introduced to both the RCSA tool tip receptance prediction and

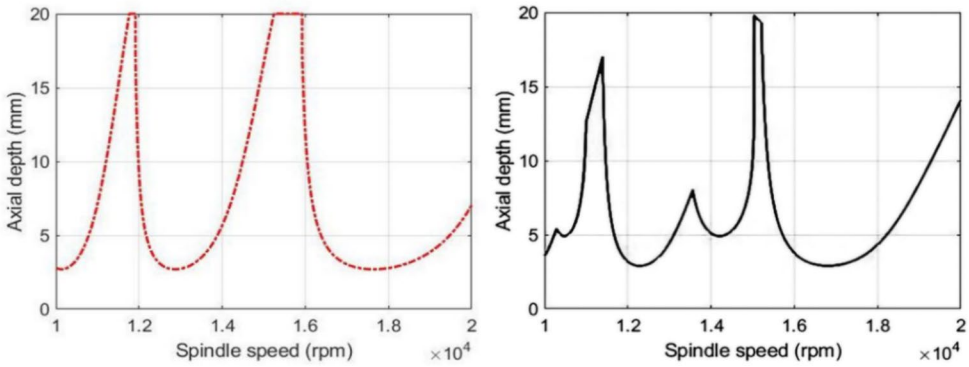

Fig. 5 Comparison of SLD with errors (*left*) and true SLD (*right*)

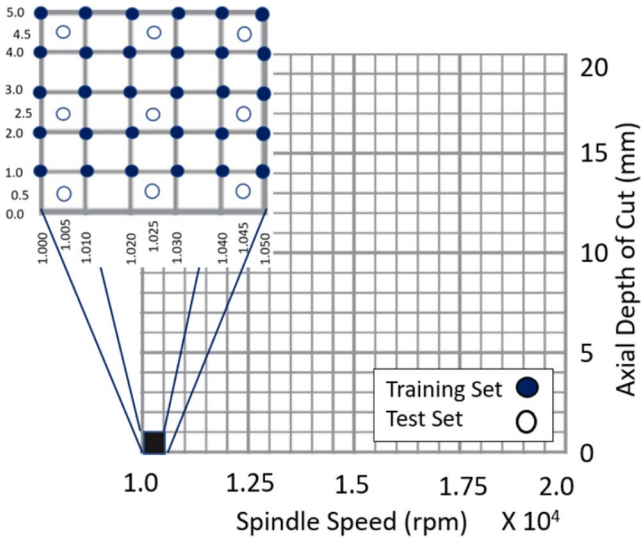

Fig. 6 Grid Design for Simulation Experiments

the force model to incorporate uncertainty. These errors include the extension length of the tool inserted in the holder for the RCSA model and the cutting force coefficients for the force model. The SLD is defined with and without these errors, where the SLD provides the limiting axial depth-of-cut as a function of spindle speed for a given radial depth-of-cut, milling orientation, and number of endmill teeth.

The two SLDs generated for the following experiments are shown in Fig. 5. The SLD on the left has been generated from the physics-based stability model with introduced errors; the true SLD on the right has been generated from the physics-based stability model without errors. As can be seen, there is a wide divergence between the SLD with errors and the SLD without errors (i.e., the true SLD). For this exercise, the correct values for the milling setup are: (1) 12 mm diameter tool with 4 teeth and a 53 mm extension length from the holder; and (2) a force model with coefficients $k_{tc} = 692.8$ N/mm^2, $k_{nc} = 400.0$ N/mm^2, $k_{te} = 0$, and $k_{ne} = 0$. These parameters were used to define the right SLD in Fig. 5.

Because there is uncertainty in these inputs, errors were introduced to define the starting SLD displayed on the left in Fig. 5. Errors were introduced for the tool extension length and force model coefficients. These erroneous values were: (1) tool with 50 mm extension from the holder; and (2) force model coefficients $k_{tc} = 649.0$ N/mm^2, $k_{nc} = 262.2$ N/mm^2, $k_{te} = 0$, and $k_{ne} = 0$. The same spindle and holder receptances were used in both cases and the same stability algorithm was applied (with modifications to the tool geometry and force model). Up milling with a radial depth of 2 mm was selected.

Research framework

Simulation experiments were conducted over a 2-dimensional grid of 2,020 points described by spindle speed and axial depth-of-cut space as illustrated in Fig. 6. Spindle speeds ranged from 1.0×10^4 to 2.0×10^4 rpm along the horizontal x-axis, in increments of 100.0 rpm; axial depth-of-cut values ranged from 0 to 20 mm along the vertical y-axis, in increments of 1.0 mm. The training dataset is shown as solid circles. For testing the PGML models, another dataset of points was created using the midpoints of alternate x-axis and y-axis intervals in the training grid, also illustrated in Fig. 6. This out-of-sample test set includes 500 points shown by the open circles in the figure. Each combination of axial depth-of-cut and spindle speed in both the training and test sets is associated with two physics-based stability values (physics-based or true) computed as described in Sect. 5.

For baseline training of the PGML model prior to updating, each spindle speed and axial depth-of-cut point in the training set of 2,020 points is associated with a physics-based stability value. During training, the PGML model is "fit" to this baseline training dataset of 2,020 points using three different machine learning methods and then tested against the test set of 500 points. To update the PGML model, the true stability states of measured points are sampled from a set of true stability values, computed as previously described, and their values updated in the PGML training dataset. In

Table 1 K-Nearest Neighbors Hyperparameters and Value Ranges

K-NEAREST NEIGHBORS	
Hyperparameter	***Value Range***
# K-neighbors	1–50, step = 2
Weights	Uniform, Distance
Metric	Euclidean, Manhattan
# Stratified CV Folds	5
Scoring	Accuracy

Selected: K = 3, weights = uniform, metric = Euclidean, CV = 5, scoring = accuracy

practice, as discussed earlier, the true states are unknown. For the purposed of this simulation, true stability values are defined by the physics-based model without errors, as described earlier. Thus, for each combination of spindle speed and axial depth-of-cut in both the training data set of 2,020 points and the test dataset of 500 points, there are two associated stability states, the simulated physics-based value and the simulated true stability value. Over the sequence of iterations, as more points are sampled, measured, and updated, the baseline PGML training dataset contains fewer physics-based stability values and an increasing number of true or validated stability values.

Training the physics-guided machine learning model

To evaluate the performance of the proposed approach, PGML models for each updating strategy were implemented using three machine learning methods: (1) K-Nearest Neighbors (KNN), (2) Support Vector Machines (SVM), and (3) Artificial Neural Networks with Nesterov-Accelerated Adaptive Moment Estimation (ANN-NADAM). Each reflects a different learning approach that captures different characteristics of the stability data being modeled. Four updating strategies are evaluated: (1) Random Sample without Domain Knowledge (RAN), (2) Random Sample with Domain Knowledge (RAN-DK), (3) Climb-the-Hill with Domain Knowledge (CTH-DK), and (4) Follow-the-Curve with Local Search and Domain Knowledge (FTC-LS-DK). In total, 12 experimental designs were created.

K-Nearest Neighbors (KNN)

KNNs (Tran et al., 2021; Deshmukh & Bhosle, 2018) are a non-parametric supervised machine learning method used largely for classification but also regression. Also known as "lazy learner", KNN makes decisions by referring to the K data points closest to the data point of interest. The distance between any two data points is calculated by using any of several metrics. Common choices are Manhattan, Euclidean, and Minkowski distances. KNN (1) does not make strict assumptions over the distribution of the dataset, unlike linear regression; (2) performs especially well under the infinite sample set assumption, with less probability of error than any other decision rule; and (3) does not produce a generalized rule over the dataset, which means it requires less training time—an advantage for in-process sampling and measurement. However, it stores all training data during the validation phase which requires more memory. Hyperparameters for the KNN model are provided in Table 1.

Table 2 Support Vector Machine Hyperparameters and Value Ranges

SUPPORT VECTOR MACHINE	
Hyperparameter	***Value Range***
Kernels	RBF, Linear
C range	[1, 1.25, 1.5, 1.75, 2, 3, 5, 10, 25, 50, 100, 1000]
Gamma range	[0.00001, 0.0001, 0.001, 0.01, 0.1, 1, 10, 100, 1000]
Stratified CV Folds	5

Selected: Kernel = RBF, C = 1000, Gamma = 1.0, CV = 5, scoring = accuracy

Support Vector Machine (SVM)

SVM (Peng et al., 2015; Wan et al., 2021) is a supervised machine learning method that is used largely for classification, but also prediction. Like ANNs, SVMs infer a function from labeled training data consisting of a set of training examples of paired inputs and outputs. The objective of the SVM algorithm is to find a hyperplane in an N-dimensional space where N is the number of input features that distinctly classifies the data points. For example, binary classification is performed by finding the hyperplane that best differentiates between two classes, i.e. maximizes the margin between the hyperplane and the support vectors, or closest values to the classification margins. The use of kernels can transform linearly inseparable problems into linearly separable ones. Given the highly nonlinear SLD, SVM's ability to transform linearly inseparable problems may be an advantage. Hyperparameters for the SVM model are provided in Table 2.

Artificial neural network with Nesterov-Accelerated Adaptive Moment Estimation (NADAM)

NADAM is a type of gradient descent optimizer used in neural network models that minimizes the cost function by finding the optimized values for the weights during updating. NADAM is typically used in the case of noisy gradients or gradients with high curvatures. One limitation of gradient descent is that a single learning rate (i.e. step size) is used for all input variables. Extensions to gradient descent like the Adaptive Moment Estimation (ADAM) algorithm add a first and second moment of the gradient and automatically adapt the learning rate for each parameter that is being optimized. This approach may result in a step size that rapidly decreases to very small values. NADAM is an extension of ADAM that incorporates Nesterov momentum (NAG) and

Table 3 Artificial Neural Network with NADAM Hyperparameters and Value Ranges

ARTIFICIAL NEURAL NETWORK WITH NADAM	
Hyperparameter	***Value Range***
# Input Features	2
# Output Targets	1
ANN Configuration	[20-14-8-1]
# Hidden Layers	3
# Neurons in Hidden Layers	[20-14-8]
# Neurons in Output Layer	1
Hidden Layer Activation Function	ReLu
Output Layer Activation Function	Sigmoid
Loss Function	Binary Crossentropy
Optimizer	NADAM
Metric	Accuracy
# Epochs for Training	1000

can result in better performance of the optimization algorithm. NAG is an extension to classical momentum where the update is performed using the gradient of the projected update to the parameter rather than the actual current variable value. This has the effect of slowing down the search when the optimum is located rather than overshooting. Thus, ANN-NADAM may have an advantage when fitting SLDs with large numbers of lobes and associated optima. Recent applications of ANN to machining include chatter prediction in milling (Mishra & Singh, 2022) and a forward prediction model for electrochemical micro-machining with knowledge embedded into an ANN (Lu et al., 2017). Hyperparameters for the ANN model are provided in Table 3.

Updating the physics-guided machine learning stability model

The PGML model is updated with simulated true stability states that would be measured in an operational environment, as described earlier. At each iteration, the replacement of the baseline physics-based (and uncertain) stability values with true stability values will force convergence of the physics-based SLD towards the true SLD for that particular milling machine, operational environment, and status of tool wear. By updating the PGML model with measured points, augmented by domain knowledge points, the process by which measurement data is captured on the factory floor can be simulated. An important question, then, is how to sample points for measurement to promote fast and efficient convergence of the PGML SLD to the true underlying SLD—and thus achieve the highest predictive accuracy with the fewest measured points.

Four updating strategies are implemented, as described and illustrated below, to evaluate the speed and efficiency of convergence of the PGML SLD to the true SLD at each iteration. The dashed red line represents the physics-based simulated PGML SLD, while the solid black line is the true simulated SLD. For the purpose of determining efficiency of convergence, we distinguish between points that are measured—points that are, in practice, experimentally determined during production and for which there is an acquisition cost in terms of production interruption or downtime—and points that do not have any associated cost because they are determined by domain knowledge rather than experiment. Points can be measured or updated only once to avoid double counting although there is no cost associated with updating a domain knowledge point. All measured points are updated unless they have been updated by domain knowledge in a previous iteration.

Random sample without domain knowledge (RAN)

Random Sample without Domain Knowledge assumes no *a priori* theory-based or expert domain knowledge to guide selection of measured points and, thus, provides a benchmark against which the information value of prior knowledge can be evaluated. Points are sampled randomly across the experimental grid without replacement. No points are updated by domain knowledge. To start the updating process, 1,000 points are randomly sampled from the grid. At each of 10 update iterations, 100 points are randomly selected from the initial sample set of 1,000 points. The true stability states of these points are measured and their respective values are updated in the PGML training dataset. When all 100 sample points in each iteration have been measured and updated, the PGML model is retrained. Figure 7 illustrates the RAN updating strategy for a small set of 13 randomly sampled points. In the left-hand figure, solid red points that lie above the PGML SLD, shown as a red dashed line, are unstable prior to updating. Conversely, solid green points below the PGML SLD are stable prior to updating. After measuring the true states of all 13 points, four points have been updated, as shown in the grid on the right. Two unstable points have been updated to stable, indicated by the arrows and light green outlined points; similarly, two stable points have been updated to unstable, indicated by arrows and light red outlined points. No points have been updated by domain knowledge. After 10 iterations, all 1,000 sampled points have been measured and their stability states updated in the PGML training dataset.

Random sample with domain knowledge (RAN-DK)

Random Sample with Domain Knowledge leverages domain knowledge, or physics-based knowledge, to update additional grid points to their *true* value, but without measurement. The addition of domain knowledge points increases

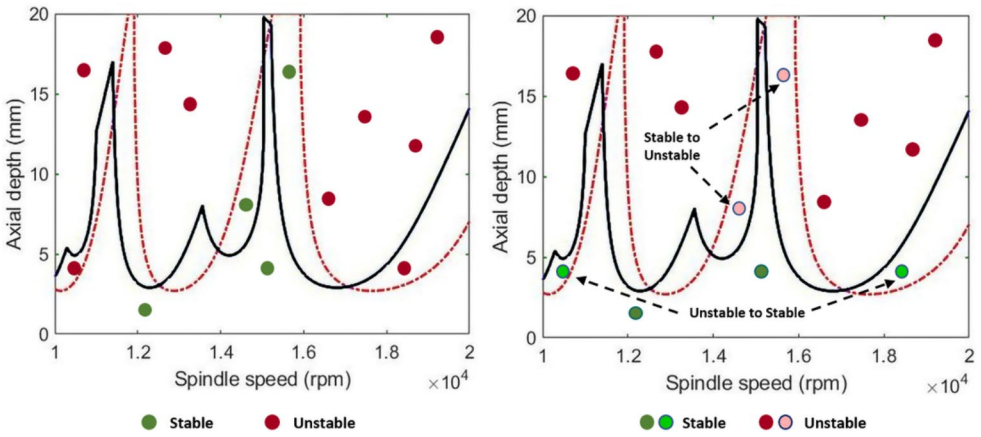

Fig. 7 Random Sample. Sampled points at first iteration (*left*) and measured and updated points (*right*)

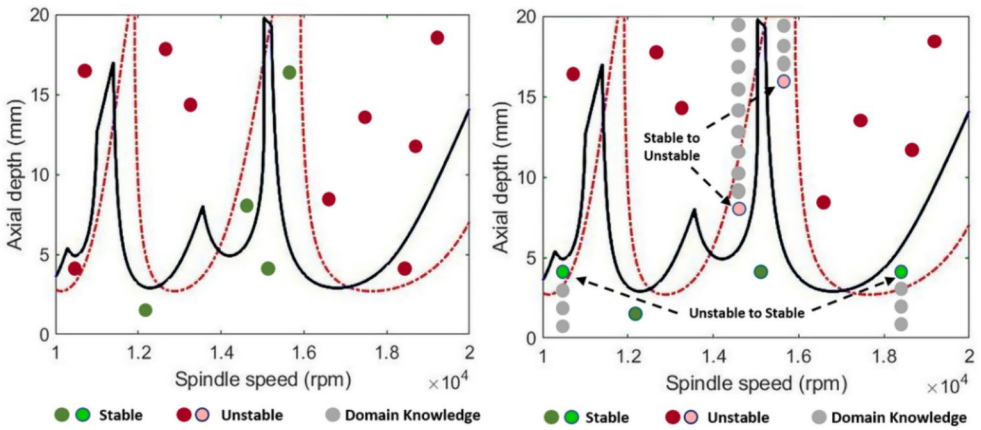

Fig. 8 Random Sample with Domain Knowledge. Measured stable and unstable points (*left*) and updated domain knowledge points (*right*)

the information content of the training set and speeds convergence to the true SLD. It is known from theory (and practice) that all values of axial depth-of-cut above those that have been measured to be unstable must also be unstable—assuming constant spindle speed. Similarly, all points below the axial depth-of-cut of a measured stable point must also be stable—assuming constant spindle speed. The addition of domain knowledge points also reduces the number of points that need to be measured to achieve target convergence. In Fig. 8, the domain knowledge points that have been updated to stable below a measured stable point—or unstable above a measured unstable point—are shown in light gray without an outline.

Climb-the-Hill with domain knowledge (CTH-DK)

Climb-the-Hill with Domain Knowledge leverages a strategy that discovers the unknown (in practice) underlying SLD by systematically measuring points at the lowest depth-of-cut and incrementing upward until chatter is observed. The process is analogous to how many manufacturers develop their guidelines for parameter selection. The process is initiated by measuring and updating all points with axial depth-of-cut equal to 1.0 mm and spindle speeds ranging from 1.0×10^4 to 2.0×10^4 rpm. At each subsequent iteration, as samples "climb-the-hill," the axial depth-of-cut is incremented one unit. In the following experiments, each incremental unit equals 2 mm. All points at the incremented depth-of-cut that lie above a point measured to be stable in the previous iteration are measured and updated. Further, since all points above a measured unstable point are unstable, all points above a measured unstable point are also updated as unstable in the PGML training dataset through domain knowledge. As illustrated in Fig. 9, on the left, all points at 1.0 mm axial depth-of-cut are measured to be stable during the first iteration and are shown in solid dark green. Therefore, at the second iteration on the right, all points are measured at axial depth-of-cut equal to 3.0 mm. During the second iteration, five points are measured to be unstable and are shown outlined in light red. Their values are updated to unstable in the PGML training dataset, along with all points above it by domain knowledge. Domain knowledge points are shown in light gray. Similarly, at the third iteration at axial depth-of-cut of 5.0 mm, only points above a stable point in the previous iteration are measured. Seven additional points are measured as unstable and their values are updated to unstable in the PGML training dataset, along

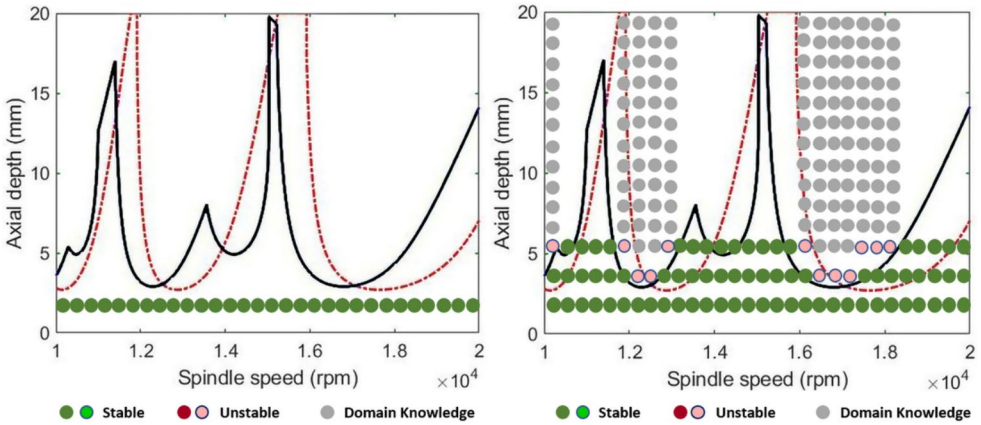

Fig. 9 Climb-the-Hill. Measured stable points at first iteration (*left*) and updated measured and domain knowledge points at second and third iterations (*right*)

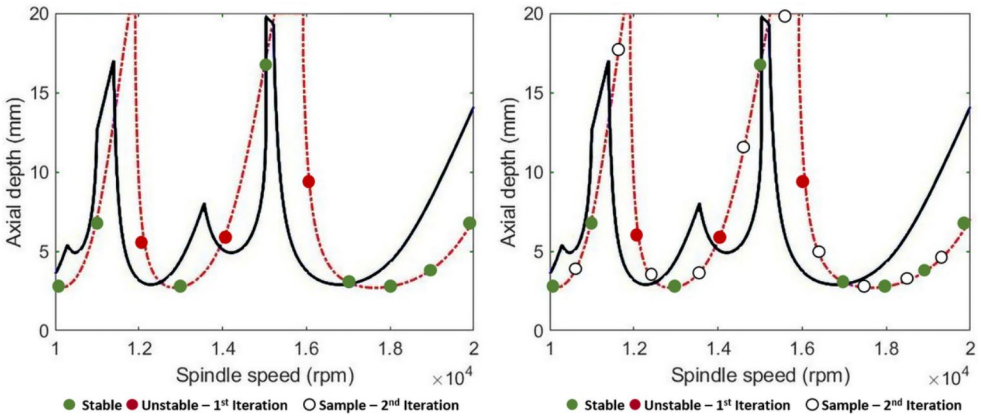

Fig. 10 Follow-the-Curve. Sampled points at first iteration (*left*) and at second iteration (*right*)

with all domain knowledge points above them. The process continues similarly at each iteration.

Follow-the-Curve with local search and domain knowledge (FTC-LS-DK)

Follow-the-Curve with Local Search and Domain Knowledge guides the process of selecting points for measurement along the physics-based SLD where the information value of the measured points is hypothesized to be highest due to its likely proximity to the true underlying SLD. The selection strategy is completed by domain knowledge and a local search process which also focuses measurement in the area of highest information value. Points are measured at equal intervals of spindle speed along the baseline PGML SLD and in increasingly smaller increments of spindle speed at each iteration. Follow-the-Curve also includes a local search routine that measures points around the sample points in a prescribed way. The strength of the Follow-the-Curve strategy is that the local search allows it to gain more information, more quickly, about the location of the true SLD. Domain knowledge is implemented for both measured sample and for local search measured points around the sample points on the SLD.

As shown in Fig. 10, Follow-the-Curve begins by selecting points at regular intervals of spindle speed along the x-axis. The sample point to be measured is the point on the bsaeline PGML SLD associated with the selected value of spindle speed on the x-axis. For example, at the first iteration, 11 points are selected at equal increments of 1.0×10^3 rpm along the spindle speed axis from 1.0×10^4 to 2.0×10^4 rpm. The corresponding points on the PGML SLD are initially assumed to be stable. After measurement three points, shown in red, are unstable while eight points, shown in green, are confirmed stable. Measured values are updated in the PGML training dataset. At each subsequent iteration, midpoints for each of the spindle speed intervals on the x-axis, shown as outlined white circles on the right, are measured and updated, as necessary. The process continues until a final interval size of 100 rpm at Iteration 6, at which point every spindle speed value along the x-axis in the training dataset has been measured.

After each sample point on the baseline PGML SLD is measured, a local search is implemented around that point. Adjacent points in each of the four compass directions (north, south, east and west) are sequentially measured and their values updated. The local search in each direction stops when the measured state matches the value in the PGML training dataset. Local search is illustrated in Fig. 11.

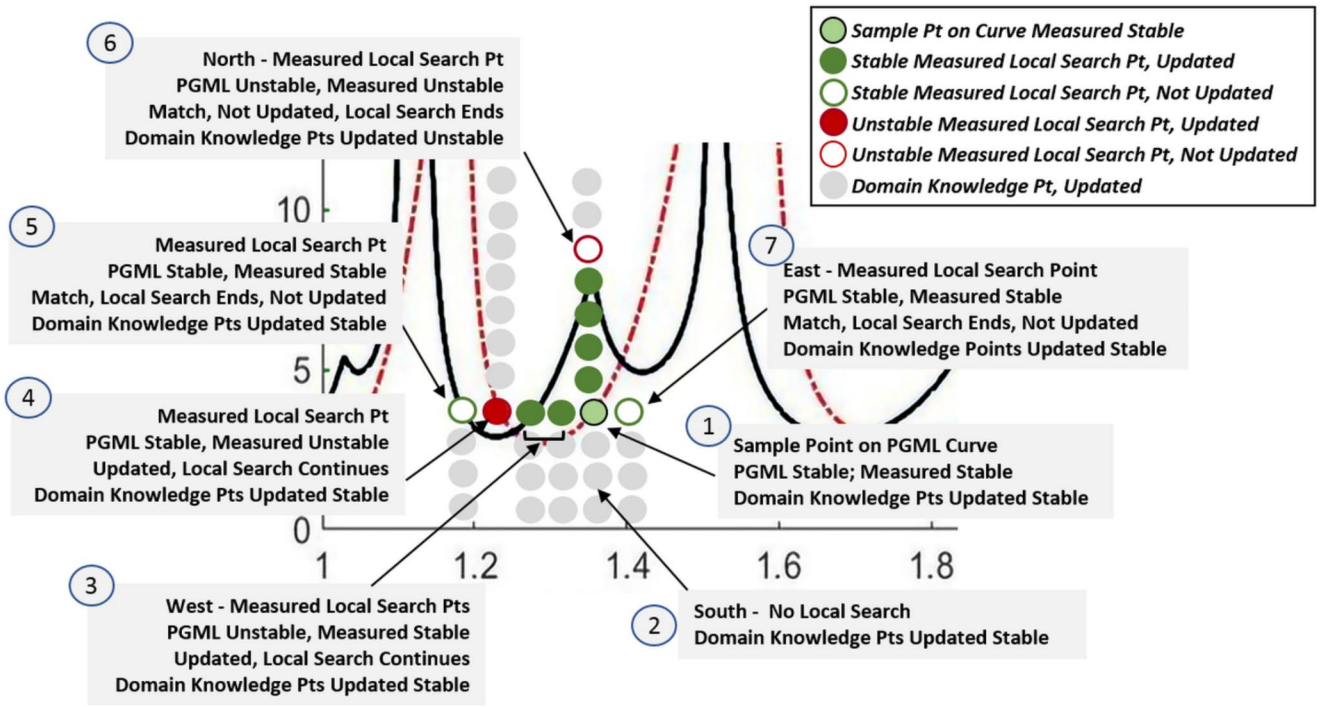

Fig. 11 Follow-the-Curve with Local Search and Domain Knowledge

Following the numbering, at Step 1 the sample point on the curve at spindle speed 1.4×10^4 rpm, indicated as a solid light green circle, is measured as stable and its stability value in the PGML training dataset is updated. At Step 2, proceeding in the south direction, all points below the measured stable point are updated as stable by domain knowledge without measurement. In all cases, if a sample point is measured to be stable, no further measurement is performed in the south direction and all points below are updated as stable without measurement. Similarly, no further measurement is performed in the north direction if a sample point is measured to be unstable and all points above that point are updated as unstable without measurement. Domain knowledge points are updated above or below the sampled point, but never in both directions.

Continuing with Step 3, to the west we measure the next adjacent point as stable which does not match the value in the PGML training dataset, and its value is updated in the PGML training dataset. All points below that point are updated as stable without measurement by domain knowledge and the local search to the west continues. An additional point is measured. Its stability state does not match the value in the PGML training dataset and is also updated. In Step 4, the next point measured is unstable and does not match the value in the PGML training dataset, indicated by a solid red circle. All points above that point are updated as unstable without measurement by domain knowledge and the local search to the west continues. Finally, at Step 5, the measured local search point to the west is measured as stable which matches the value in the PGML training dataset and the local search in that direction stops. Domain knowledge points below that point are updated as stable. Search in the north (Step 6) and east (Step 7) directions proceed similarly.

Experimental design for training and updating the PGML model

The workflow for implementing PGML model training and updating in each of the experiments is provided in Fig. 12. Each of three machine learning methods was paired with each of the four updating methods and metrics for evaluating their respective predictive accuracies and stability convergence scores according to the performance metrics described below. As illustrated in the flowchart in Fig. 12, three physics-based models are used to analytically determine the physics-based dynamic stability model and its associated SLD. This model informs the creation of two simulated datasets. The first is the physics-based dataset for baseline training of the PGML and the second is the true dataset from which the measurement values are sampled (simulating experimental data collection on the factory shop floor) for updating the PGML model. Values of these datasets populate a 2-dimensional experimental grid over the experimental domain of interest where the x-axis is spindle speed (rpm) and the y-axis is depth-of-cut (mm).

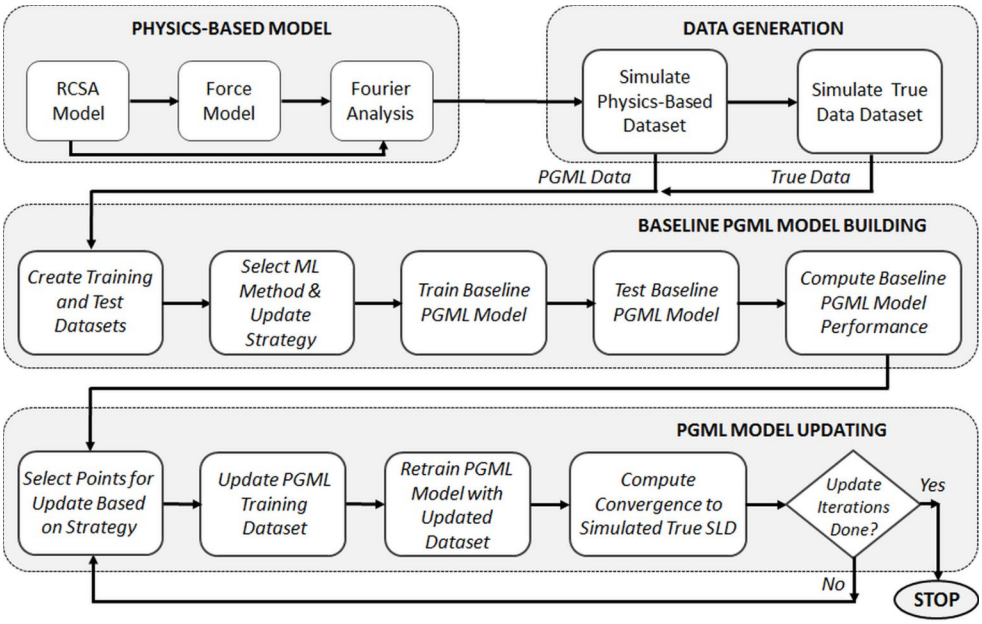

Fig. 12 Process for Training, Testing and Updating PGML Model at Each Iteration

A baseline PGML model is fit to the initial physics-based training dataset and predictions of the stability states are produced for the test set. Model accuracy and stability convergence are computed to serve as a benchmark for evaluating the convergence of the updated PGML models to the true underlying SLD. In an iterative fashion, measurements of true stability states are sampled from the simulated true dataset as specified by the updating strategy and the PGML dataset is updated. At each iteration, the stability states are predicted for the test set and performance metrics are computed to track convergence of the physics-based SLD to the true underlying SLD.

Performance metrics A_{TRAIN} and A_{TEST} express the percent of the points for which the predicted PGML stability value matches the stability value in the training dataset, and the percent of points for which the predicted PGML stability value matches the stability value in the test dataset, respectively. Training and testing accuracy are computed in the conventional way. Training accuracy, A_{TRAIN}, is defined as the number of points in the *training set* for which a correct prediction (CP) is obtained by the PGML model when compared with the training dataset values, divided by the total number of predictions (P). Testing accuracy, A_{TEST}, is defined as the number of points in the *test set* for which a correct prediction (CP) is obtained by the PGML model when compared with the test dataset values, divided by the total number of predictions (P). The general expression of accuracy, for both training and testing, with respect to their training and test datasets, is provided in Eq. 5 below:

$$\mathrm{A_{TRAIN}}(\%) = \mathrm{A_{TEST}}(\%) = \frac{CP}{P} * 100 \tag{5}$$

Stability convergence, C_{SLD}, expresses the percent of points for which the PGML-predicted stability value matches the true stability state at each iteration. Changes in the magnitude of C_{SLD} from iteration to iteration reflect the speed with which the PGML SLD gets closer and closer to, or converges to, the *true* SLD. A high stability convergence value means that the PGML model agrees more closely with the true stability model and that the number of correct stability predictions by the PGML model is high. Similarly, increasing convergence scores at each iteration indicate that the PGML SLD curve is converging towards the *true* SLD curve and that the predictive capability of the PGML model is increasing. Stability convergence C_{SLD} is formally defined as the number of points in the *test set* for which a correct prediction (CP) is obtained by the PGML model when compared with the *true value*, divided by the total number of predictions (P), as follows in Eq 6:

$$\mathrm{C_{SLD}}(\%) = \frac{CP}{P} * 100 \tag{6}$$

F_1Score. The F_1 score, also called F-measure, is the harmonic mean between recall and precision where precision is the number of true positive (TP) results determined by the model divided by the number of all positive (TP + FP) results, and recall is the number of true positive (TP) results divided by the number of all samples that should have been identified as positive (TP + FN). The F1 score takes both precision and recall into account at equal weights and generates

Table 4 Performance Results for Baseline PGML Model Fitting

Machine Learning Model	Average Accuracy		Average F1 Score		Average AUC Score		Average Convergence
	Train	Test	Train	Test	Train	Test	Fit to True Model
KNN	100.0%	96.7%	99.9%	95.5%	100.0%	99.7%	81.0%
SVM	99.6%	97.0%	99.4%	95.9%	99.0%	100.0%	81.2%
ANN-NADAM	97.2%	95.6%	95.6%	93.8%	99.6%	99.3%	81.5%

a high score only if the number of true positives obtained is high compared to the other prediction outcome categories. The F_1 score, computed as shown in Eq. 7, ranges from 0.0 to +1.0 (expressed as a percentage) where a higher score indicates a better classifier performance.

$$F_1 = 2 * \frac{\text{precision} * \text{recall}}{\text{precision} + \text{recall}} = \frac{2 * TP}{2 * TP + FP + FN} \quad (7)$$

Area Under the Curve (AUC). The AUC measures the ability of a classifier to distinguish between classes. The AUC equals the probability that the classifier will rank a randomly chosen positive example higher than that of a randomly chosen negative example. The AUC, ranging between 0.0 and +1.0 (expressed as a percentage), is typically used for binary classification problems. The higher the AUC, the better the model is at distinguishing between classes.

In addition to computing performance metrics A_{TRAIN}, A_{TEST}, C_{SLD}, F_1 and AUC, the number of measured points and domain knowledge points that are validated as stable or unstable at each iteration are also counted. Tracking point counts, especially counts of measured points that incur cost, provides insight into the information value of measurement and domain knowledge with respect to predictive accuracy. Depending on updating strategy, different numbers and locations of points will be measured and updated at each iteration. Similarly, different numbers and locations of domain knowledge points will be confirmed as stable or unstable at each iteration. These different spatial patterns of sampling and updating for each strategy drive improvements in convergence to the true SLD and make a particular updating strategy better than another from the perspective of predictive accuracy and measurement cost.

Numerical results

Numerical results addressing the following research questions are provided below: (1) Does the baseline PGML model, without updating, adequately *fit* the physics-based data, and which machine learning method provides the best fit as a baseline for subsequent updating; (2) Can the predictive ability of the baseline PGML model be improved by updating the PGML training dataset with measured data and domain knowledge, and which updating strategy provides the best convergence (i.e. predictive accuracy); and (3) Can predictive accuracy be further improved using physics-based knowledge of the SLD by implementing local search around measured points. Results for each of the 12 experiments are presented. Results from each experiment are averages of 100 runs. Reference to the physics-based and true SLDs in the following discussion of experimental results refers to the simulated SLDs, respectively.

Accuracy and convergence during baseline PGML model fitting

A first question is whether the baseline PGML model, without updating, adequately *fits* the physics-based data and which machine learning method provides the best fit. To compare methods, the baseline PGML model was trained using KNN, SVM and ANN-NADAM and metrics A_{TRAIN} and A_{TEST} were computed. Typically, A_{TRAIN} results are higher than A_{TEST} results because the points in the test dataset have not been evaluated or seen by the PGML model during training. Note that model *fit* here refers to the fit with the baseline physics-based dataset and does not reflect how well the PGML model predicts the true stability states. To evaluate how well the PGML model predicts true stability states, stability convergence C_{SLD} is computed. Both accuracy and convergence results can help assess which of the machine learning methods will perform best when retraining the PGML model during updating.

Table 4 compares the training and testing accuracies for the three machine learning methods KNN, SVM, and ANN-NADAM, as well as two other useful metrics for classification problems with class imbalance, F_1 and AUC scores. KNN provides the best training accuracy, correctly predicting all stability values in the training dataset. All methods, however, exhibit A_{TRAIN} scores of 97.2% or higher. Performance against the test dataset is comparably good, with A_{TEST} scores between 95.6% and 97.0% for all methods. Thus, all machine learning methods provide overall good fit with the physics-based model and presage good predictive accuracy during updating. Stability convergence C_{SLD} results are also provided in Table 4 and range between 81.0% and 81.5%. ANN-NADAM achieved the highest C_{SLD} score at 81.5%. All three methods provide a comparable baseline for evaluating predictive accuracy improvements due

Table 5 Stability Convergence and Point Counts for Random (RAN)

RANDOM									
Iteration	Sample Size	KNN C_{SLD}	SVM C_{SLD}	ANN-NADAM C_{SLD}	Measured Points	Domain Knowledge Points	Validated Points	Cum. Validated Points	Grid Points
Baseline		81.0%	81.2%	81.5%					
1	100	81.3%	81.8%	82.4%	100	0	100	100	5.0%
2	100	81.6%	82.2%	83.0%	100	0	100	200	9.9%
3	100	81.9%	81.6%	83.8%	100	0	100	300	14.9%
4	100	82.0%	82.0%	84.3%	100	0	100	400	19.8%
5	100	83.1%	81.2%	85.4%	100	0	100	500	24.8%
6	100	83.9%	82.4%	85.9%	100	0	100	600	29.7%
7	100	85.1%	83.2%	86.6%	100	0	100	700	34.7%
8	100	86.0%	84.6%	87.5%	100	0	100	800	39.6%
9	100	87.1%	86.6%	88.4%	100	0	100	900	44.6%
10	100	88.3%	87.8%	89.0%	100	0	100	1000	49.5%
%Change		9.0%	8.1%	9.2%	1000	0	1000		

to measurement and domain knowledge updating and were retained for further experiments.

However, to account for small differences in performance across the methods during updating, each updating strategy is paired with each machine learning method in12 experiments, as noted earlier. Interestingly, ANN-NADAM performed least well among the three methods with respect to both A_{TRAIN} and A_{TEST}, but slightly better with respect to C_{SLD}. Average F_1 scores and AUC scores are also presented for each method in Table 4. Beyond accuracy, the F_1 score takes into account not only the number of prediction errors made by a model, but also captures the type of errors that are made. A model will obtain a high F_1 score only if it can predict most of the positive, chatter cases (i.e. high precision) and if it can correctly identify most of the positive, chatter cases (i.e. high recall) in the data. As shown in Table 4, the methods have high F_1 score varying from 93.8 to 95.9% for the test dataset. The higher the AUC score the better the ability of the method to distinguish between the positive and negative classes, i.e. chatter and no-chatter. The three methods' AUC scores exhibit very good performance varying from 99.3 to 100% on the test dataset.

Convergence during PGML model updating

A second question is whether the predictive ability of the baseline PGML model can be improved by updating the PGML training dataset with operational data and domain knowledge, and which updating strategy provides the best convergence (i.e. predictive accuracy). Having confirmed that each machine learning method offers a good fit to the physics-based training dataset, and comparable stability convergence, all 12 experiments were performed for the four updating strategies described in Sect. 6.0 and all performance metrics computed for both training and testing. The stability convergence scores and point counts for each iteration are provided in Tables 5, 6, 7 and 8 for all experiments. Accuracy, F_1 and AUC scores, computed for each iteration over all experiments, were as expected and are omitted due to space limitations.

Stability convergence, C_{SLD}, is expected to increase as the PGML training dataset is updated with measured values and domain knowledge at each iteration consistent with the hypothesis that the PGML model is converging to the true stability model. Stability convergence, as noted earlier, reflects the increasing accuracy of the PGML model in predicting the underlying true stability values. This hypothesis is confirmed by the results in Tables 5, 6, 7 and 8. Conversely, and as expected, as stability values in the PGML training dataset are updated with measured values, stability convergence C_{SLD} improves for all 12 experiments. Again, looking at KNN for the random updating strategy (RAN) in Table 5, C_{SLD} increases from 81.0% at the baseline iteration to 88.3%, or an increase of 9.0% and ANN increases from 81.5 to 89.0% or an increase of 9.2%. Similar increases in C_{SLD} from baseline iteration to final iteration are obtained for the other three updating strategies as shown in Tables 6, 7 and 8. Stability convergence is defined as the percent of correct stability predictions achieved during testing. All four strategies achieved a predictive accuracy of at least 88%. The Follow-the-Curve strategy achieved the highest predictive accuracy of 95% with KNN.

The comparative gains in predictive accuracy by PGML models over physics-based models (equivalent to baseline PGML model) are summarized in Fig. 13. The results are as expected but contain some surprises. RAN, the benchmark against which other updating strategies can be compared, had the lowest C_{SLD} improvement of 8.8% averaged over the three machine learning methods. RAN can be thought of as an uninformed method since no *a priori* knowledge

Table 6 Stability Convergence and Point Counts for Random with Domain Knowledge (RAN-DK)

RANDOM WITH DOMAIN KNOWLEDGE									
Iteration	Sample Size	KNN	SVM	ANN-NADAM	Measured Points	Domain Knowledge Points	Validated Points	Cum. Validated Points	Grid Points
		C_{SLD}	C_{SLD}	C_{SLD}					
Baseline		81.0%	81.2%	81.5%					
1	100	84.8%	84.6%	85.1%	85	203	288	288	14.3%
2	100	86.8%	85.2%	86.2%	72	175	247	535	26.5%
3	100	88.3%	88.8%	88.4%	49	205	254	789	39.1%
4	100	89.1%	89.2%	89.5%	48	128	176	965	47.8%
5	100	90.9%	92.0%	90.9%	25	93	118	1083	53.6%
6	100	92.0%	92.6%	91.3%	37	38	75	1158	57.3%
7	100	92.3%	92.6%	91.6%	36	103	139	1297	64.2%
8	100	93.1%	93.2%	92.0%	22	87	109	1406	69.6%
9	100	93.6%	94.0%	91.9%	27	64	91	1497	74.1%
10	100	93.7%	94.4%	92.2%	9	45	54	1551	76.8%
%Change		15.7%	16.3%	13.1%	410	1141	1551		

Table 7 Stability Convergence and Point Counts for Climb-the-Hill (CTH-DK)

CLIMB-THE-HILL									
Iteration	Climbing Increment	KNN	SVM	ANN-NADAM	Measured Points	Domain Knowledge Points	Validated Points	Cum. Validated Points	% of Grid Points
		C_{SLD}	C_{SLD}	C_{SLD}					
Baseline		81.0%	81.2%	81.5%					
1	1 mm	81.0%	81.2%	81.4%	101	0	101	101	5.0%
2	3 mm	81.8%	81.6%	82.6%	101	199	300	401	19.9%
3	5 mm	87.8%	88.0%	86.7%	91	298	389	790	39.1%
4	7 mm	91.6%	91.8%	90.4%	53	194	247	1037	51.3%
5	9 mm	94.0%	92.8%	92.3%	29	121	150	1187	58.8%
6	11 mm	94.9%	94.0%	93.0%	19	108	127	1314	65.0%
7	13 mm	95.1%	94.8%	93.0%	14	67	81	1395	69.1%
8	15 mm	95.0%	94.6%	93.0%	8	28	36	1431	70.8%
9	17 mm	95.0%	94.6%	93.1%	4	0	4	1435	71.0%
10	19 mm	94.9%	94.6%	93.2%	2	18	20	1455	72.0%
%Change		17.2%	16.5%	14.4%	422	1033	1455		

Table 8 Stability Convergence and Point Counts for Follow-the-Curve (FTC-LS-DK)

FOLLOW-THE-CURVE									
Iteration	Omega Step	KNN	SVM	ANN-NADAM	Measured Points	Domain Knowledge Points	Validated Points	Cum. Validated Points	% of Grid Points
		C_{SLD}	C_{SLD}	C_{SLD}					
Baseline		81.0%	81.2%	81.5%					
1	1000	89.2%	88.0%	87.9%	109	360	469	469	23.2%
2	800	91.0%	91.0%	90.3%	63	143	206	675	33.4%
3	600	92.7%	93.4%	91.8%	45	37	82	757	37.5%
4	400	94.0%	93.8%	92.4%	28	89	117	874	43.3%
5	200	94.5%	94.6%	92.9%	48	34	82	956	47.3%
6	100	95.0%	94.8%	92.9%	39	13	52	1008	49.9%
%Change		17.3%	16.8%	14.0%	332	676	1008		

is used to sample points for measurement and no domain knowledge is implemented. In contrast, RAN-DK, CTH-DK, and FTC-LS-DK all implement physics-based theory and/or domain knowledge in different ways. While neither RAN-DK nor CTH-DK assumes a *priori* knowledge of the physics-based SLD, they do utilize domain knowledge to increase predictive accuracy. FTC-LS-DK implements *a priori* knowledge of the physics-based PGML SLD to guide sampling directly on the physics-based SLD curve. FTC-LS-DK also has the benefit of local search. Local search is hypothesized to add information about the location of the true underlying SLD by additional measurement in

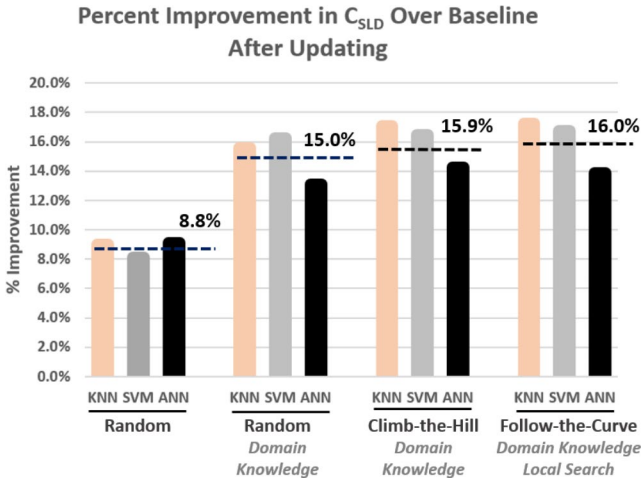

Fig. 13 Percent Improvement in Stability Convergence Over Baseline After Updating

close proximity to the true SLD. RAN-DK, which builds in domain knowledge, increased the percent improvement in C_{SLD} over baseline to 15.0%. Surprisingly, RAN-DK, CTH-DK and FTC-LS-DK obtain equivalent results, as seen in Fig. 13. All three strategies improved C_{SLD} over baseline by (on average) 15.0%, 15.9% and 16.0%, respectively, after a complete iteration cycle. Local search did not add substantially to improvements in C_{SLD} when comparing FTC-LS-DK and CTH-DK strategies.

More important, while RAN-DK, CTH-DK and FTC-LS-DK produced comparable improvement in C_{SLD} over their respective complete iteration cycles, the trajectories of improvement at each iteration are very different. Iterative improvements in C_{SLD} for the four updating strategies are compared in Fig. 14. As can be observed in the figure, each strategy has a unique signature. Recognizing that the preferred updating strategy is the one that maximizes C_{SLD} with minimal investment in measurement, the desired signature is a rapid increase in C_{SLD} at early iterations when cumulative measurement costs are still low. As expected, RAN underperforms the other three strategies, achieving less than 90% convergence after 10 iterations. When equal numbers of points are selected at random without *a priori* theoretical knowledge, improvements from iteration to iteration can be expected to be approximately uniform, as observed by the linear signature. Any variability across ML methods for RAN can be explained by different spatial positioning of the measured points with respect to the PGML SLD at each of 100 repetitions of the experiment. RAN-DK approaches 95% stability convergence by the 10th iteration which can be explained by the influence of additional domain knowledge points. Declining C_{SLD} improvement for RAN-DK at later iterations reflects fewer numbers of measured points since points in the sample to be measured have been already updated by domain knowledge at previous iterations.

In particular, very different behaviors can be observed for the CTH-DK and FTC-LS-DK strategies. While both achieve C_{SLD} of 95%, the trajectories are quite different. With CTH-DK, stability convergence starts off slowly for the first two iterations and then increases rapidly before flattening after the fifth iteration. In contrast, FTC-LS-DK stability convergence increases sharply from baseline to the first iteration, then shows smaller increases at subsequent iterations. As reported in Tables 5, 6, 7 and 8, and illustrated graphically in Fig. 13, the percent increase in C_{SLD} across the iteration cycles ranges between 8.1% and 9.2% for RAN, between 13.1% and 16.3% for RAN-DK, between

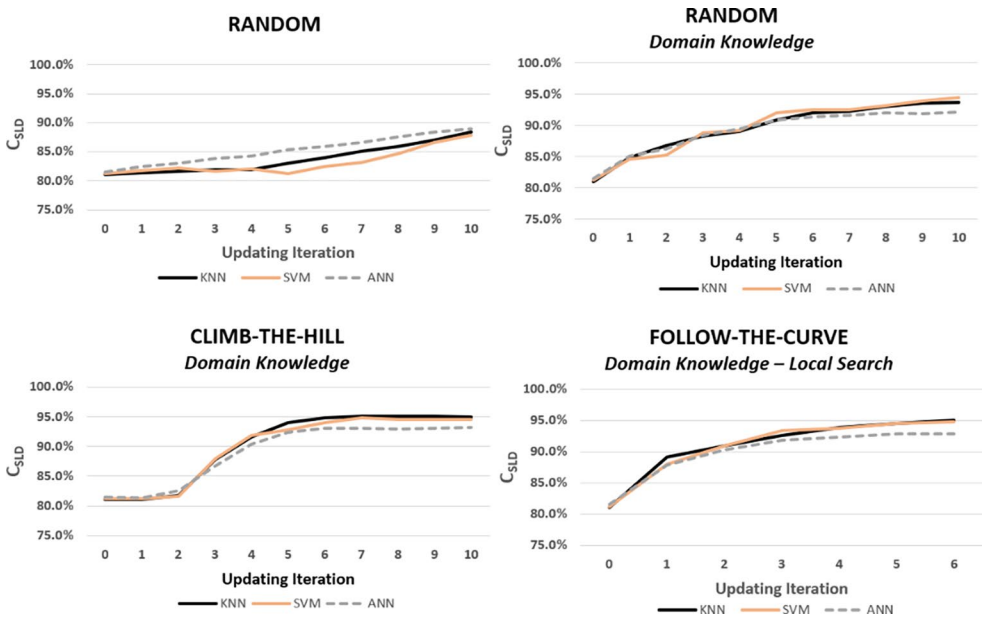

Fig. 14 Stability Convergence C_{SLD} by Updating Strategy and Iteration

14.4% and 17.2% for CTH-DK, and between 14.0% and 17.3% for FTC-LS-DK.

Convergence of the PGML SLD to the true underlying SLD at each iteration can be visualized graphically for FTC-LS-DK in Fig. 15. The true SLD is shown by a solid black line and the PGML SLD by a dashed red line. The darkly shaded areas between the true SLD and the PGML SLD, as well as the lightly shaded gap areas below the true SLD are regions of prediction error. The dark pink shaded areas indicate regions where unstable stability states are incorrectly predicted as stable by the PGML. The light pink shaded areas below the true SLD define regions in which the stability states are correctly predicted as stable; similarly, the white areas above both curves define regions where unstable states are correctly predicted as unstable. And lightly shaded areas between the two curves define regions where stable states are incorrectly predicted as unstable. At the baseline iteration a number of regions of prediction error, noted by dark shaded regions above the true SLD and lightly shaded regions below the true SLD, can be seen. As more measured points are updated in the training dataset, these regions decrease in size. By the last iteration, the updated PGML SLD has substantially converged to the true SLD with a few exceptions, e.g. in particular, the large area to the right side of the figure at high spindle speeds and a small lobe to the left size of the figure at low spindle speeds.

Local search stopping rules for Follow-the-Curve

To further improve the performance of the Follow-the-Curve strategy (FTC-LS-DK), three stopping rules that control the pattern local search around sampled points on the physics-based SLD were investigated. Local search gathers information of higher value in areas that are hypothesized to be close to the true underlying SLD. As described in Sect. 6, each local search point, is measured and updated in the four compass directions until the process is halted by the stopping rule in effect. For each measured local search point, domain knowledge points are also updated, without measurement. Stopping Rule 1 was implemented in the previous experiments.

> ***Stopping Rule 1: Stop on Match***. *Local search in each compass direction stops when the measured value equals the value in the PGML training dataset and the local search measured point is updated. Domain knowledge points are updated above or below the measured local search points, as appropriate.*
>
> ***Stopping Rule 2: Stop on Mismatch***. *Local search in each compass direction stops when the measured value does* ***not*** *equal the value in PGML training dataset and the local search measured point is updated. Domain knowledge points are updated above or below the measured local search points, as appropriate.*
>
> ***Stopping Rule 3: Stop on Measured Sample Point Mismatch***. *The local search in each compass direction stops when the measured value does not equal the stability value of the measured sample point (the local search original starting point) and the local search measured point is updated. Domain knowledge points are updated above or below the measured local search points, as appropriate.*

Figure 16 illustrates the different search behaviors of each stopping rule and allows visual comparison of the relative performance of each with respect to convergence. Graphs in the left column illustrate local search mechanics for Stopping Rule 1, Stopping Rule 2, and Stopping Rule 3, all at iteration 1. Graphs in the right column compare the convergence of the final PGML SLD to the true SLD after all six iterations. Stopping Rule 1 offers the best performance with fewer regions of prediction error after the last iteration. As previously, sample measured points are outlined in red; local search measured points in the four compass directions are shown in solid green; domain knowledge points are shown in light gray. Also, as previously, the baseline PGML model prior to Iteration 1 was trained using the physics-based training dataset. At subsequent iterations, the PGML model was retrained with updated training datasets. ANN-NADAM was used to train the PGML model at all iterations—the best performing of the three methods during baseline testing.

The mechanics of local search are easily seen in the top left in Fig. 16. Stopping Rule 1 measures points in each compass direction until the measured local search point matches the value in the PGML training dataset. Sample point A, indicated in the figure by the shaded red circle, is predicted by the baseline PGML model to be stable, but is measured in Iteration 1 as unstable. Since the point is measured to be unstable, all points to the north are updated as unstable, as shown. Moving to the east, the point to the right is measured as unstable, which matches the PGML model value and the local search in that direction ends. Next, a local search in the south direction is conducted measuring stable points until it hits Point B which measures stable, as predicted by the PGML model. Finally, local search turns to the west direction, measuring stable points until the measured value is equal to PGML predicted value of stable which stops the local search to the west.

The superior performance of Stopping Rule 1, and the reason for its selection as the local search stopping rule, can be seen by comparing the regions of prediction error in the graphs on the right. All three stopping rules result in regions of prediction error on the far right and left of the graph.

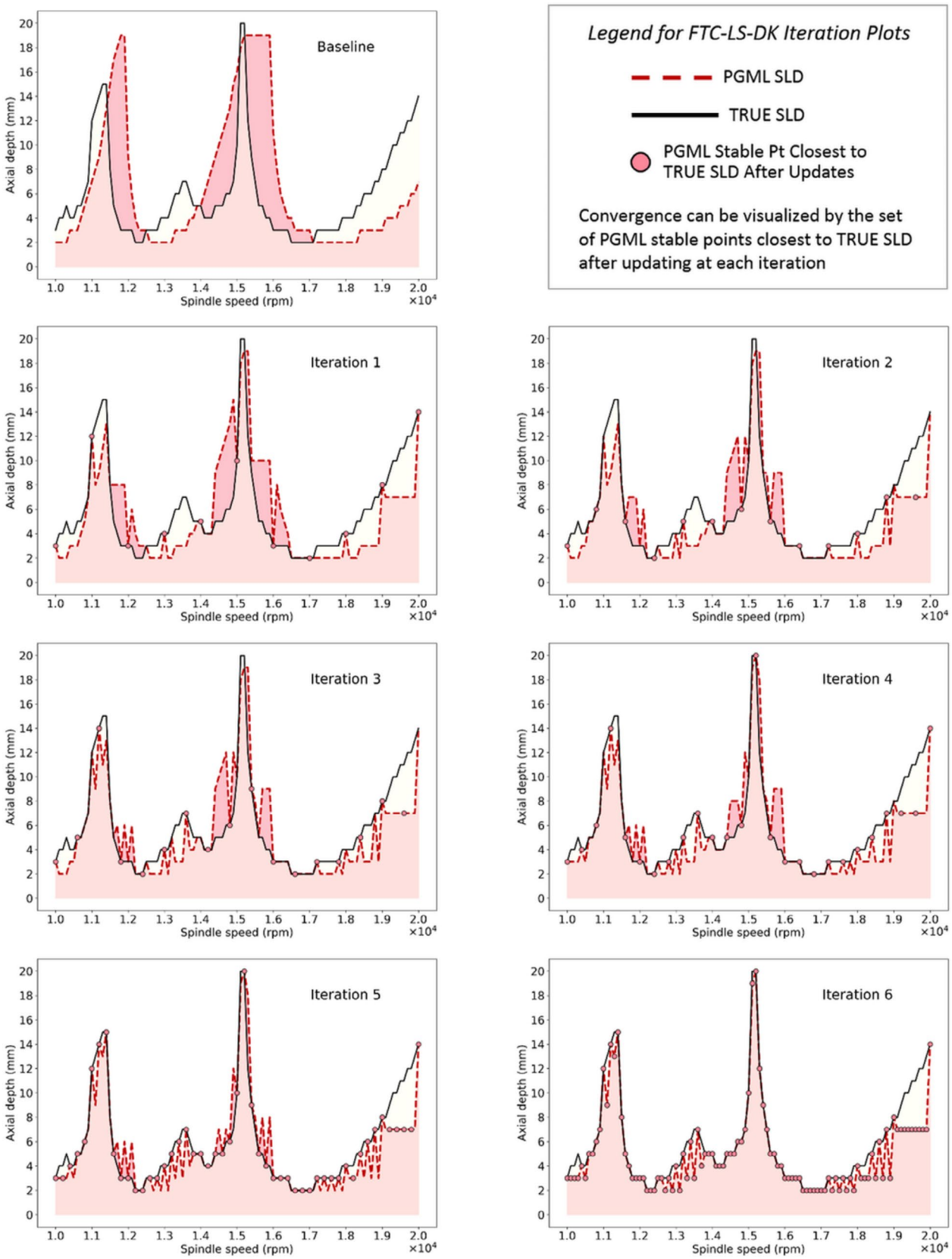

Fig. 15 Convergence of PGML SLD to True SLD with FTC-LS-DK Updating Strategy

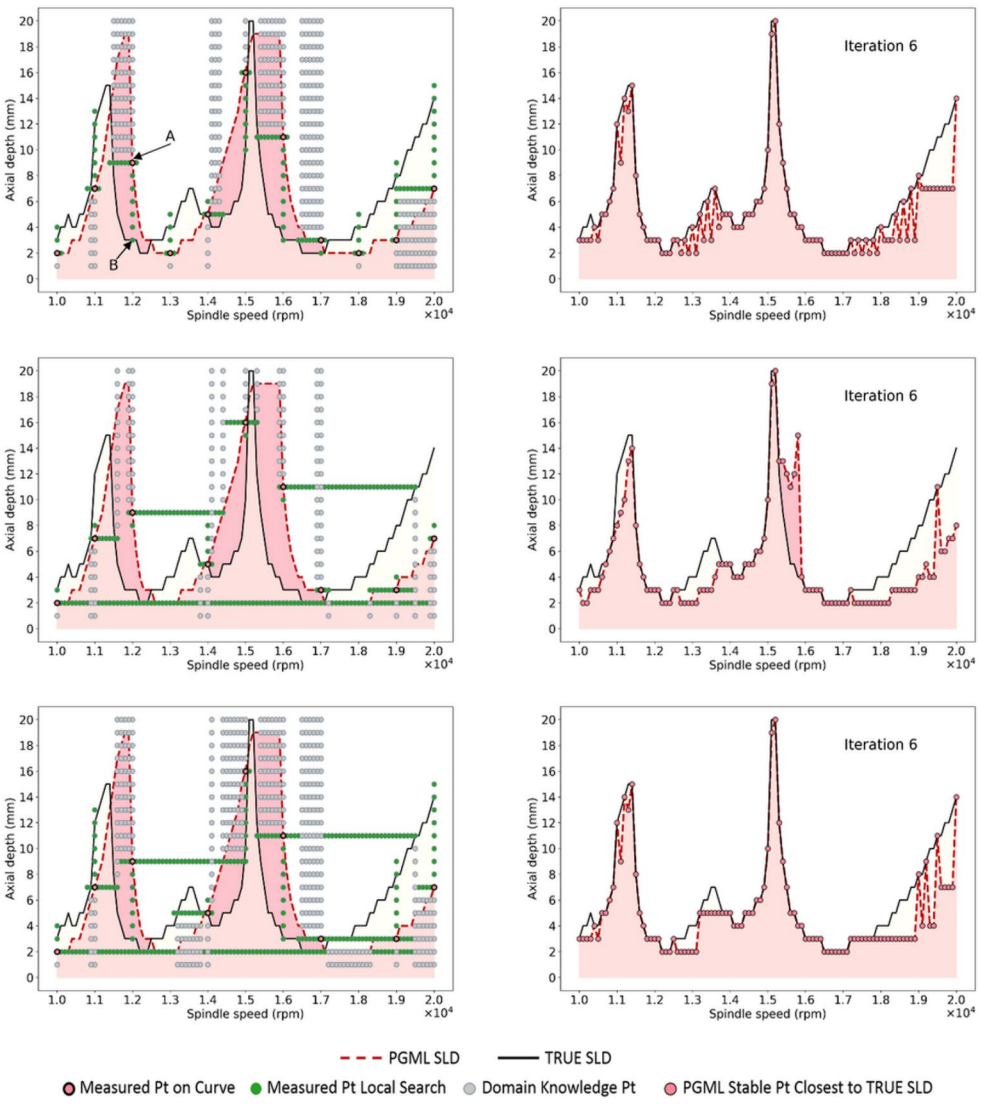

Fig. 16 Comparison of Local Search Stopping Rule Behaviors for FTC-DK: Rule 1 (*top row*), Rule 2 (*middle row*), and Rule 3 (*bottom row*)

However, Stopping Rule 2 fails to correctly predict stability in three additional regions, two of which are around the tall stability lobes. In two of the three regions the PGLM model predicts unstable states when the true states are stable while, in the third, the PGML model predicts stable states when the true states are unstable. Stopping Rule 3 reduces prediction errors around the tall lobes, but the third region of prediction error is only slightly reduced.

Numerical results in Table 9 comparing accuracy, A_{TRAIN} and A_{TEST}, and convergence C_{SLD} for the three stopping rules are consistent with the above observations. With respect to $C_{SLD,}$ Stopping Rule 1 and Stopping Rule 3 achieve 92.1% and 91.5% convergence, respectively, both higher than Stopping Rule 2 at 89.4%. When differences in baseline fit for each stopping rule are considered, Stopping Rule 1 and Stopping Rule 3 report the highest improvements of C_{SLD} over baseline of 12.9% and 12.1%, respectively. Stopping Rule 2 reports only 9.6% over baseline, respectively.

The superior performance of Stopping Rule 1, as well as Stopping Rule 3 can be explained by the larger numbers of grid points that are validated as true by measurement or domain knowledge. A reasonable expectation is that a stopping rule that maximizes the number of measured points that are updated, as well as domain knowledge points, would have an advantage. The number of points validated as to their true state by measurement and domain knowledge, shown in Table 10, also support the superior performance of Stopping Rule 1 and Stopping Rule 3. Table 10 compares the total points validated for each stopping rule over all six iterations, broken down into measured points (both on the curve and by local search) and unmeasured domain knowledge points. As shown in Table 10, Stopping Rule 1 achieves the highest convergence (92.1%) with the fewest measured points (332) and the most domain knowledge points (676), for a total of 1008 validated points. Stopping Rule 3 achieves comparable convergence (91.5%) with

Table 9 Accuracy and Convergence for Local Search Stopping Rules (FTC-LS-DK with ANN-NADAM)

	LOCAL SEARCH STOPPING RULE 1				LOCAL SEARCH STOPPING RULE 2				LOCAL SEARCH STOPPING RULE 3			
Iteration	A_{TRAIN}	A_{TEST}	C_{SLD}	True Points	A_{TRAIN}	A_{TEST}	C_{SLD}	True Points	A_{TRAIN}	A_{TEST}	C_{SLD}	True Points
Baseline	96.8%	95.1%	81.6%	79.6%	96.8%	95.1%	81.6%	79.6%	96.8%	95.1%	81.6%	79.6%
1	93.3%	87.2%	87.8%	88.6%	94.6%	92.1%	84.1%	82.6%	92.3%	85.4%	88.7%	89.7%
2	92.4%	84.5%	90.0%	91.2%	93.8%	90.7%	84.4%	84.3%	92.5%	83.7%	90.2%	92.0%
3	91.8%	83.8%	90.5%	92.3%	93.9%	89.6%	85.5%	85.9%	92.5%	83.1%	90.6%	93.3%
4	92.4%	83.1%	91.1%	93.4%	93.8%	88.1%	86.9%	87.5%	92.3%	82.7%	90.9%	94.1%
5	92.2%	82.2%	91.6%	94.7%	93.8%	86.2%	88.7%	89.8%	92.3%	82.4%	91.0%	94.7%
6	92.8%	80.6%	92.1%	96.1%	94.3%	84.8%	89.4%	91.6%	92.8%	81.5%	91.5%	95.6%
%Change	-4.1	-15.3%	12.9%	20.7%	-2.6	-10.8%	9.6%	15.1%	-4.1	-14.3%	12.1%	20.1%

432 measured points and 641 domain knowledge points, for a total of 1073 validated points. Stopping Rule 2 has the fewest validated points (962) and, as observed earlier, the domain knowledge points are not in locations that provide leverage to reduce the size of the regions of prediction error.

Another factor contributing to the superior performance of Stopping Rule 1 is the percent of total points in the PGML dataset that have been updated to their true state at the end of a complete iteration cycle. A particular stopping rule may, by chance, validate more points than another stopping rule that change their state to the true stability value since they sample different sets of points. However, at the end of the iteration cycle, nearly all grid points are expected to have been updated to their true value through measurement or domain knowledge. To confirm this, the number of true points as a percent of total points in the grid, is shown in Table 9. At the end of the iteration cycle, the percentages of validated true points for Stopping Rules 1, 2 and 3 are 96.1%, 91.6%, and 95.6%, respectively.

Balancing predictive accuracy and measurement cost in the factory

Measuring the stability states for combinations of spindle speed and axial depth-of-cut can be prohibitively expensive for many machine tool companies and job shops. In a production environment, keeping manufacturing costs low is a priority. And while avoiding chatter both improves surface quality and reduces the expense of rework and waste, the costs and time needed to implement the PGML approach in practice can be sizeable and must also be considered. From a practical perspective, then, an important consideration for factory floor operations is selection of an updating strategy that balances predictive accuracy (i.e. stability convergence) and measurement cost. Assessing this trade-off is a major goal of this research.

Each of the four updating strategies investigated here offers a different approach for gathering measurement data during machining in order to avoid chatter. In particular, numerical experiments were designed to evaluate the ability of the PGML approach to locate the true underlying SLD with minimal process measurements. Control of the measurement process from iteration to iteration was defined by novel updating strategies that update points through measurement, physics-based theory or domain knowledge. Since the numbers and locations of updated points vary by updating strategy at each iteration, it is difficult to compare strategies directly by looking at stability convergence $\mathbf{C_{SLD}}$ and measured point counts separately. While these metrics tell us a lot about the predictive accuracy of the PGML model overall, they do not provide practical insight for

Table 10 Point Counts for Local Search Stopping Rules (FTC-LS-DK with ANN-NADAM)

LOCAL SEARCH STOPPING RULE	SAMPLE PTS ON CURVE	PTS MEASURED ON CURVE	PTS MEASURED LOCAL SEARCH	TOTAL MEASURED PTS	DOMAIN KNOWLEDGE PTS	TOTAL VALIDATED PTS
Rule 1	219	54	278	332	676	1008
Rule 2	219	67	346	413	549	962
Rule 3	219	38	394	432	641	1073

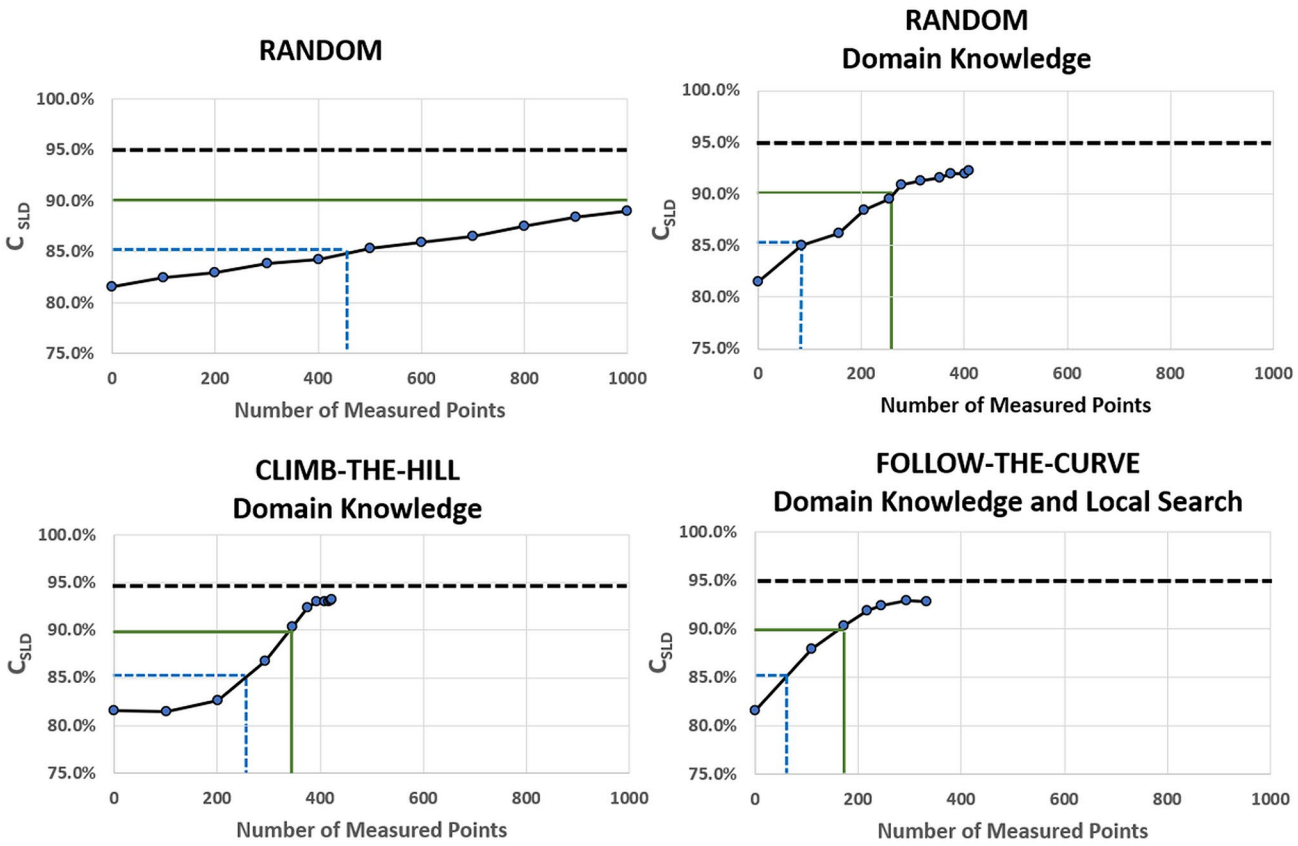

Fig. 17 Trade-Off Between Predictive Accuracy and Measurement Cost – ANN-NADAM

applications that depend on the information efficiency—or the information contribution of each measured point to gains in predictive accuracy. For decisions on the factory floor, then, the critical question is which and how many points should be measured to quickly locate the true position of the SLD. That is, what is the marginal value of measuring an additional point to the convergence of the PGML SLD to the true SLD. And additionally, how many points need to be measured to achieve a *target* stability convergence—that is, how many points need to be measured if one wants to be 95% accurate in their predictions of dynamic stability.

This trade-off between stability convergence C_{SLD} and number of measured points is provided in Fig. 17 for each updating strategy using ANN-NADAM for model fitting and updating. Referring to Fig. 17, consider first a target of 95% stability convergence shown by the dashed black line. The number of measured points required to approach 95% stability convergence varies greatly by strategy. The RAN updating strategy never reaches 95% stability convergence after 1,000 measured points over 10 iterations. After 10 iterations, approximately 50% of the grid has been validated by measurement but only 89.0% convergence is achieved. RAN-DK, CTH-DK and FTC-LS-DK approach 95% target accuracy with significantly fewer measured points, plus contributions by domain knowledge points. The addition of (unmeasured and no cost) domain knowledge points to the updating strategy enables 92.2% convergence after 410 measured points over 10 iterations for RAN-DK, 93.2% convergence for CTH-DK after 10 iterations with 422 measured points, and 92.9% convergence for FTC-LS-DK after only 322 measured points in 6 iterations. The trade-off between stability convergence C_{SLD} and number of

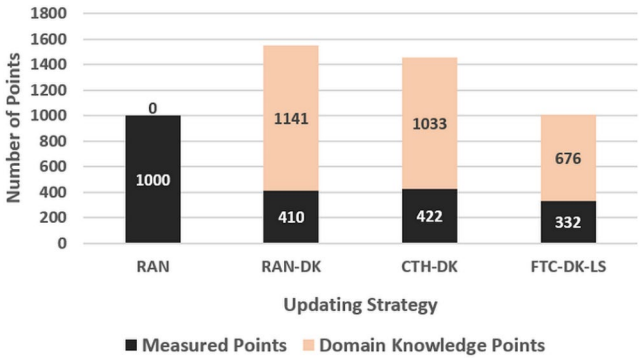

Fig. 18 Points Validated by Measurement and Domain Knowledge per Updating Strategy (All Iterations)

measured points with target accuracy of 90% is also shown in Fig. 17 by a solid green line. Again, FTC-LS-DK is the most efficient updating strategy from a measurement cost perspective after 6 iterations, achieving 90% convergence with only 180 measured points. Results for a target accuracy of 85% are also shown in Fig. 17 by a dashed blue line.

From a practical perspective, the updating strategy of choice would provide the most improvement in C_{SLD} with the fewest points—and also the most quickly. The speed of convergence can be seen in the shapes of the curves in Fig. 17. The updating strategies display different geometric behaviors as C_{SLD} increases. The RAN and RAN-DK strategies display an approximately linear relationship between C_{SLD} and the number of measured points, as expected since each iteration starts with a selection of 100 randomly sampled points, with RAN-DK measuring fewer points at later iterations. The CTH-DK strategy displays a convex relationship between C_{SLD} and the number of measured points, with small increases in C_{SLD} at early iterations even though larger numbers of points are measured. The FTC-LS-DK strategy, in contrast, displays a concave relationship between C_{SLD} and the number of measured points. Stability convergence, C_{SLD}, for FTC-LS-DK increases more quickly during early iterations with fewer measured points overall but including local search points with high information value in close proximity to the true SLD—an advantage on the factory floor where low costs are a priority.

Finally, the contribution of domain knowledge, in addition to measured points discussed above, helps to explain the comparative performance of the updating strategies. Figure 18 summarizes the total numbers of measured and domain knowledge points over all iterations for the four updating strategies. By design, RAN does not leverage domain knowledge. However, each of the three other updating strategies benefits from the contribution of domain knowledge to the predictive accuracy of the PGML model, and the faster convergence of the PGML SLD to the true SLD. The superior performance of FTC-LS-DK, discussed earlier, can be attributed to the contributions of both measured points and domain knowledge points located spatially in the region of the true SLD due to the sampling strategy that is guided by the physics-based model.

Conclusions

This research applies a new approach, physics-guided machine learning or PGML, to modeling dynamic stability for process control to avoid chatter during milling. The simulation experiments performed here address the problem of estimating the true—and unknown—stability boundary or SLD by updating the physics-based stability model with physics-based theory to guide the sampling strategy, combined with measurement data and domain knowledge. The PGML approach proposed offers the ability to improve productivity and part quality by selecting operational parameters that avoid chatter while minimizing production downtime required for data collection.

Simulation experiments were performed in which the PGML stability model was trained using an initial approximation of the physics-based model (with errors) and then updated with simulated measurement data (without errors) to approximate the true stability model specific to the operational environment. Four strategies to update the PGML model were explored—each strategy reflecting a different approach to incorporating both physics-based theory, real-time measurements and domain knowledge. All four updating strategies improved the predictive capability of the baseline physics-based model, with improvements ranging between 8.1% and 17.3%, depending on the ML method used, to achieve prediction accuracy approaching 95%. Updating strategies that leveraged both the information value of each measured and updated point and non-measured domain knowledge points, achieved the largest gains in predictive accuracy and converged most quickly to the underlying true stability boundary.

Further, all four updating strategies demonstrated convergence to the true underlying SLD, even when measurement points were selected randomly. The inclusion of domain knowledge in addition to measurement data for the Random with Domain Knowledge (RAN-DK), Climb-the-Hill with Domain Knowledge (CTH-DK) and Follow-the-Curve with Domain Knowledge and Local Search (FTC-LS-DK) strategies resulted in further improvements in both the magnitude and speed of convergence in all experiments—demonstrating the value of domain knowledge to data-driven stability modeling. Further, the choice of updating strategy mattered less than the fact that domain knowledge was included. Among the three updating strategies with domain knowledge, all demonstrated comparable convergence

capability—approximately 15% improvement in convergence to the true SLD.

Follow-the-Curve (FTC-LS-DK) achieved the fastest convergence with the fewest measured points—suggesting its suitability for use in a production environment where measurement costs are a concern. Follow-the-Curve (FTC-LS-DK) leveraged physics-based knowledge about the approximate location of the true underlying SLD to obtain a faster rate of convergence at early iterations. A practical limitation of FTC-DK-LS is that *a priori* knowledge of the physics-based SLD is required to guide the selection of points to be sampled for measurement. Neither RAN-DK and CTH-DK nor require *a priori* knowledge of the physics-based SLD yet achieved comparable predictive accuracy, although with more measurement cost. Finally, the RAN-DK, CTH-DK and FTC-LS-DK strategies all approached 95% convergence to the true SLD over their respective complete iteration cycles. The proposed methods are currently being validated with experiments using real data obtained during the milling process in a laboratory setting. Work is continuing to refine the updating strategies reported here to further improve chatter prediction and reduce measurement cost. In particular, in-process knowledge of chatter occurrence while milling can allow adaptive updating strategies for choosing the next points for measurement. In addition, other ML methods such as hybrid models (Deshmukh & Bhosle, 2018) are being explored for more efficient baseline training.

The PGML approach shows promise in bridging the gap between theory and practice in chatter avoidance. Despite technical advances in process control for machining processes, managing dynamic instabilities during machining remains a challenge and an impediment to increased productivity and more consistent surface quality for machined parts. The true stability state and its SLD are unknown in practice, and machine operators typically resort to experience and/or manufacturer's recommendations in setting the *feeds and speeds* to avoid chatter. Further, when chatter is experienced, the operator typically stops production to adjust operating parameters adding time and cost—and may need to remove the workpiece for later rework depending on the degree of surface roughness. The approach introduced here offers foundational support for future self-aware machining where a machine's ability to discover its own SLD allows self-adjustment independent of the operator. This capability depends on self-knowledge of both the machine's true and current stability states, plus a mapping of process, operational, and machining parameters to stable and unstable stability states during operation that allow it to maintain a stable state or return to a stable state in the event of chatter. Challenges that remain, and that are the next steps for this research, are continued development of the PGML approach and updating strategies for milling within a time-varying and continuous learning environment with the goal of enabling the capture, characterization, and prediction of the machining process in real time for machine-initiated chatter avoidance and corrections.

Acknowledgements The authors thank the Center for Self-Aware Manufacturing and Metrology (CSAM) at the University of North Carolina at Charlotte for its support of this research which was funded under a multi-year grant from the University of North Carolina's Research Opportunities Initiative.

Author contributions Greis, Nogueira, Bhattacharya and Schmitz contributed to the study conception and design. The physics-based dynamic stability modeling and generation of the simulated data were performed by Schmitz. The physics-guided machine learning methodology was developed by Greis, Nogueira and Bhattacharya. Numerical experiments were performed by Spooner and Nogueira. Analysis of the data and results was performed by Greis and Nogueira. First draft of the manuscript was written by Greis and edited by Nogueira. All authors read and approved the final manuscript.

Data Availability No experimental data was generated in this research. All data was simulated by the simulation methods described in the paper. Simulation data is available from the corresponding author upon request.

Code Availability The code generated and analyzed during the current study is not publicly available due to the fact that it constitutes an excerpt of research in progress.

Statements and declarations

Conflicts of interest/Competing interests The authors declare that they have no known conflicts of interest or competing financial interests that influenced the work reported in this paper.

References

Altintas, Y., & Budak, E. (1995). Analytical prediction of stability lobes in milling. *Annals of the CIRP*, *44*(1), 357–362. https://doi.org/10.1016/S0007-8506(07)62342-7

Cherukuri, H., Perez-Bernabeu, E., Selles, M., & Schmitz, T. (2019). Machining chatter prediction using a data learning model. *Journal of Manufacturing and Materials Processing*, *3*(2), 45. https://doi.org/10.3390/jmmp3020045

Cornelius, A., Karandikar, J., Gomez, M., & Schmitz, T. (2021). A Bayesian Framework for Milling Stability Prediction and Reverse Parameter Identification. *Procedia Manufacturing*, *53*, 760–772. https://doi.org/10.1016/j.promfg.2021.06.073

Cuomo, S., Di Cola, V. S., Giampaolo, F., Rozza, G., Raissi, M., & Piccialli, F. (2022). Scientific Machine Learning through Physics-Informed Neural Networks: Where we are and What's next.arXiv preprint arXiv:2201.05624. https://arxiv.org/pdf/2201.05624.pdf

Deshmukh, J., & Bhosle, U. (2018). A Study of Mammogram Classification using AdaBoost with Decision Tree, KNN, SVM and Hybrid SVM-KNN as Component Classifiers. *Journal of Information Hiding and Multimedia Signal Processing*, 9(3), 548–557. Retrieved 21 April 2022 from http://bit.kuas.edu.tw/~jihmsp/2018/vol9/JIH-MSP-2018-03-004.pdf

Faghmous, J. H., & Kumar, V. (2014). A big data guide to understanding climate change: The case for theory-guided data science. *Big Data*, *2*(3), 155–163. https://doi.org/10.1089/big.2014.0026

Friedrich, J., Hinze, C., Renner, A., Verl, A., & Lechler, A. (2017). Estimation of stability lobe diagrams in milling with continuous learning algorithms. *Robotics and Computer-Integrated Manufacturing*, *43*, 124–134. https://doi.org/10.1016/j.rcim.2015.10.003

Friedrich, J., Torzewski, J., & Verl, A. (2018). Online Learning of Stability Lobe Diagrams in Milling. *Procedia CIRP*, *67*, 278–283. https://doi.org/10.1016/j.procir.2017.12.213

Greis, N., Nogueira, M., Bhattacharya, S., & Schmitz, T. (2020). Physics-guided machine learning for self-aware machining. *Amer. Assoc. for Artificial Intelligence, S*pring symposium–AI and manufacturing. Retrieved 19 December 2021 from https://aiinmanufacturing.wixsite.com/symposium/physics-guided-machine-learning-for

Karandikar, J., Zapata, R., & Schmitz, T. (2010). Incorporating stability, surface location error, tool wear, and uncertainty in the milling super diagram. *Transactions of the NAMRI/SME*, *38*, 229–236

Karandikar, J., Honeycutt, A., Smith, S., & Schmitz, T. (2020). Milling stability identification using Bayesian machine learning. *Procedia CIRP*, *93*, 1423–1428. https://doi.org/10.1016/j.procir.2020.04.022

Karpatne, A., Atluri, G., Faghmous, J. H., Steinbach, M., Banerjee, A., Ganguly, A., Shekhar, S., Samatova, N., & Kuman, V. (2017). Theory-Guided Data Science: A New Paradigm for Scientific Discovery from Data. *IEEE Transactions on Knowledge and Data Engineering*, *29*(10), 2318–2331. https://doi.org/10.1109/TKDE.2017.2720168

Karniadakis, G. E., Kevrekidis, J. G. H., Lu, L., Perdikaris, P., Wang, S., & Yang, L. (2021). Physics-informed machine learning. *Nature Review Physics*, *3*, 422–440. https://doi.org/10.1038/s42254-021-00314-5

Kim, D. H., Kim, T. J. Y., Wang, X., Kim, M., Quan, Y. J., Oh, J. W., Min, S. H., Kim, H., Bhandar, B., Yang, I., & Ahn, S. H. (2018). Smart Machining Process Using Machine Learning: A Review and Perspective on Machining Industry. *International Journal of Precision Engineering and Manufacturing Green Technology*, *5*(4), 555–568. https://doi.org/10.1007/s40684-018-0057-y

Kim, S. W., Kim, I., Lee, J., & Lee, S. (2021). Knowledge Integration into deep learning in dynamical systems: an overview and taxonomy. *Journal of Mechanical Science and Technology*, *35*(4), 1331–1342. https://doi.org/10.1007/s12206-021-0342-5

Lee, K., Huang, Y., Ji, J., & Lin, C. (2018). An Online Tool Temperature Monitoring Method Based on Physics-Guided Infrared Image Features and Artificial Neural Network for Dry Cutting. *IEEE Transactions on Automation Science & Engineering*, *15*(4), 1665–1676. https://doi.org/10.1109/TASE.2018.2826362

Lu, Y., Rajora, M., Zou, P., & Liang, S. Y. (2017). Physics-Embedded Machine Learning: Case Study with Electrochemical Micro-Machining. *Machines*, *5*(1), https://doi.org/10.3390/machines5010004

Mishra, R., & Singh, B. (2022). Prediction of milling chatter using SBLMD-ANN. *Journal of Mechanical Science and Technology*, *36*, 877–882. https://doi.org/10.1007/s12206-022-0135-5

Oleaga, I., Pardo, C., Julaika, J. J., & Bustillo, A. (2018). A machine-learning based solution for chatter prediction in heavy-duty milling machines. *Measurement*, *128*, 34–44. https://doi.org/10.1016/j.measurement.2018.06.028

Peng, C., Wang, L., & Liao, T. W. (2015). A new method for the prediction of chatter stability lobes based on dynamic cutting force simulation model and support vector machine. *Journal of Sound and Vibration*, *354*, 118–131. https://doi.org/10.1016/j.jsv.2015.06.011

Postel, M., Bugdayci, B., & Wegener, K. (2020). Ensemble transfer learning for refining stability predictions in milling using experimental stability states. *International Journal of Advanced Manufacturing Technology*, *107*, 4123–4139. https://doi.org/10.1007/s00170-020-05322-w

Raissi, M., Perdikaris, P., & Karniadakis, G. E. (2017a). Physics Informed Deep Learning (Part I): Data-driven Solutions of Nonlinear Partial Differential Equations. arXiv preprint arXiv:1711.10561. https://arxiv.org/abs/1711.10561

Raissi, M., Perdikaris, P., & Karniadakis, G. E. (2017b). Physics Informed Deep Learning (Part II): Data-driven Discovery of Nonlinear Partial Differential Equations. arXiv preprint arXiv:1711.10566. https://arxiv.org/abs/1711.10566

Raissi, M., Perdikaris, P., & Karniadakis, G. E. (2019). Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, *378*, 686–707. https://www.sciencedirect.com/science/article/pii/S0021999118307125

Roscher, R., Bohn, B., Duarte, M. F., & Garcke, J. (2020). Explainable Machine Learning for Scientific Insights and Discoveries. *Ieee Access : Practical Innovations, Open Solutions*, *8*, 42200–42216. https://doi.org/10.1109/ACCESS.2020.2976199

Rubeo, M., & Schmitz, T. (2016). Mechanistic force model coefficients: A comparison of linear regression and nonlinear optimization. *Precision Engineering*, *45*, 311–321. https://doi.org/10.1016/j.precisioneng.2016.03.008

von Rueden, L., Mayer, S., Beckh, K., Georgiev, B., Giesselbach, S., Heese, R., Kirsch, B., Walczak, M., Pfrommer, J., Pick, A., Ramamurthy, R., Garcke, J., Bauckhage, C., & Schuecker, J. (2021). Informed Machine Learning - A Taxonomy and Survey of Integrating Prior Knowledge into Learning Systems. *IEEE Transactions on Knowledge and Data Engineering*. https://doi.org/10.1109/TKDE.2021.3079836

Saadallah, A., Finkeldey, F., Morik, K., & Wiederkehr, P. (2018). Stability prediction in milling processes using a simulation-based machine learning approach. *Procedia CIRP*, *72*, 1493–1498. https://doi.org/10.1016/j.procir.2018.03.062

Schmitz, T. L., & Smith, S. (2009). *Machining Dynamics: Frequency Response to Improved Productivity*. New York, NY: Springer

Schmitz, T., & Donaldson, R. (2000). Predicting high-speed machining dynamics by substructure analysis. *Annals of the CIRP-Manufacturing Technology*, *49*(1), 303–308. https://doi.org/10.1016/S0007-8506(07)62951-5

Schmitz, T., & Duncan, G. S. (2005). Three-component receptance coupling substructure analysis for tool point dynamics prediction. *Journal of Manufacturing Science and Engineering*, *127*(4), 781–790. https://doi.org/10.1115/1.2039102

Sharp, M., Ak, R., & Hedberg, T. (2018). A survey of the advancing use and development of machine learning in smart manufacturing. *Journal of Manufacturing Systems*, 8(Part C), 70–79. https://doi.org/10.1016/j.jmsy.2018.02.004

Sheikh, R., & Jahirabadkar, S. (2018). An Insight into Theory-Guided Climate Data Science—A Literature Review. *Advances in Data*

and Information Sciences, 115–125, *Lecture Notes in Networks and Systems, 38*. https://doi.org/10.1007/978-981-10-8360-0_11

Shi, J., & Liu, C. R. (2004). The influence of material models on finite element simulation of machining. *Journal of Manufacturing Science and Engineering*, *126*(4), 849–857. https://doi.org/10.1115/1.1813473

Singh, A. P., Medida, S., & Duraisamy, K. (2017). Machine-Learning-Augmented Predictive Modeling of Turbulent Separated Flows over Airfoils. *AIAA Journal*, *55*(7), 2215–2227. https://doi.org/10.2514/1.J055595

Tao, F., Qi, Q., Liu, A., & Kusiak, K. (2018). Data-driven smart manufacturing. *Journal of Manufacturing Systems*, 48(Part C), 157–169. https://doi.org/10.1016/j.jmsy.2018.01.006

Tran, M. Q., Liu, M. K., & Elsisi, M. (2021). Effective multi-sensor data fusion for chatter detection in milling process. *ISA Transactions*. Article in press. https://doi.org/10.1016/j.isatra.2021.07.005

Unver, H. O., & Sener, B. (2021). A novel transfer learning framework for chatter detection using convolutional neural networks. *Journal of Intelligent Manufacturing*, *182*, 109689. https://doi.org/10.1007/s10845-021-01839-3

Wan, S., Li, X., Yin, Y., & Hong, J. (2021). Milling chatter detection by multi-feature fusion and Adaboost-SVM. *Mechanical Systems and Signal Processing*, *156*, 107671. https://doi.org/10.1016/j.ymssp.2021.107671

Wang, J., Ma, Y., Zhang, L., Gao, R., & Wu, D. (2018). Deep learning for smart manufacturing: Methods and applications. *Journal of Manufacturing Systems*, *48*, 144–156. https://doi.org/10.1016/j.jmsy.2018.01.003

Wang, L., Zhou, Q., & Jin, S. (2020a). Physics-guided Deep Learning for Power System State Estimation. *Journal of Modern Power Systems and Clean Energy*, *8*(4), 607–615. https://doi.org/10.35833/MPCE.2019.000565

Wang, J., Li, Y., Zhao, R., & Gao, R. (2020b). Physics-Guided Neural Network for Machining Tool Wear Prediction. *Journal of Manufacturing Systems*, *57*, 290–310. https://doi.org/10.1016/j.jmsy.2020.09.005

Willard, J., Jia, X., Xu, S., Steinbach, M., & Kumar, V. (2020). Integrating physics-based modeling with machine learning: A survey. arXiv preprint arXiv:2003.04919 1(1),1–34. https://arxiv.org/abs/2003.04919

Yesilli, M. C., Khasawneh, F. A., & Otto, A. (2020). On transfer learning for chatter detection in turning using wavelet packet transform and ensemble empirical mode decomposition. *CIRP Journal of Manufacturing Science and Technology*, *28*, 118–135. https://doi.org/10.1016/j.cirpj.2019.11.003

Yu, Y., Yao, H., & Liu, Y. (2020). Structural dynamics simulation using a novel physics-guided machine learning method. *Engineering Applications of Artificial Intelligence*, 96. https://doi.org/10.1016/j.engappai.2020.103947

Zhang, R., Liu, Y., & Sun, H. (2020). Physics-guided convolutional neural network (PhyCNN) for data-driven seismic response modeling. *Engineering Structures*, *215*, 110704. https://doi.org/10.1016/j.engstruct.2020.110704