A process optimality graph for prescriptive analytics generated using established machine learning methods

Tobias M Louw, Alexander Schulze-Hulbe, Steven M Bradshaw

Department of Chemical Engineering, Stellenbosch University, Stellenbosch, 7600, South Africa
(tmlouw@sun.ac.za)

Abstract: We present the optimality graph as a simple prescriptive analytic based on historical plant data to suggest operator actions for improved process performance. The approach enables identification of modes from plant data, generation of descriptive mode labels, and identification of drivers (actions or disturbances) for transition between modes. During online operation, the approach maps current operating conditions to a previously defined process mode and suggests actions to shift the process to a more desirable mode. Importantly, we avoid developing Yet Another Machine Learning Algorithm (YAMLA) and rely exclusively on well-established methods readily available in popular libraries in the hope that this will encourage wide-spread adoption. We demonstrate the approach on an industrial iron ore flotation data set.

Keywords: process monitoring, performance assessment, machine learning, process optimization, decision support

1. INTRODUCTION

Prescriptive analytics aims to leverage data to recommend an optimal course of action. In process engineering, this entails providing advisories that are actionable by plant personnel leading to improved plant performance (Liu et al., 2015). Our research is premised on two principles. First, industrial process operations can often be characterized by process modes covering a range of operation (Tan et al., 2011). Ideally, process modes are clearly separated in state space, but significant overlaps may exist between process modes in reality due to limited observability, process- and measurement noise, etc. Regardless, it is assumed that process operation can be classified into discrete modes. Our second premise is that plant performance may be improved by comparing current operating conditions to previously recorded operations available in a process historian and determining whether operator actions have led to improved performance in the past. The actionable advisory then simply directs plant personnel to repeat previously beneficial actions (Meyer, 2023).

Consider a process (Fig. 1) which is defined by a coordinate in the continuous state space $\mathbf{x} \in \mathbb{R}^m$. The state space maps to both the online measurement space $\mathbf{y} = h(\mathbf{x}) \in \mathbb{R}^p$, as well as some scalar key performance indicator KPI $= z(\mathbf{x}) \in \mathbb{R}$, which may be available online or only following retrospective analysis (e.g., through offline laboratory tests assessing the product quality). The process state $\mathbf{x}$ is subject to operator actions $\mathbf{u}$ as well as uncontrolled disturbances $\mathbf{d}$.

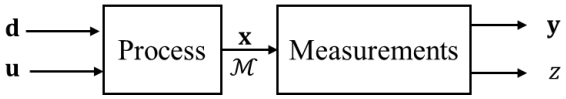

Figure 1. Operator actions $\mathbf{u}$ and uncontrolled disturbances $\mathbf{d}$ affect the process state $\mathbf{x}$ and mode $\mathcal{M}$, which is in turn observed through $\mathbf{y}$ and characterized by a (retrospectively) assigned scalar KPI z.

We assume the process can be classified into discrete process modes based on online measurement, i.e., $\mathcal{M} = f(\mathbf{y}) \in \mathbb{N}$ (we discuss this concept with more nuance in section 2.1), and that the KPI varies sufficiently from one mode to the other such that the modes can be ranked in terms of performance with a reasonable level of statistical significance. Further, mode shifts can be investigated and attributed to either operator actions $\mathbf{u}$, or disturbances $\mathbf{d}$. In this case, the process can be described by an optimality graph (Meyer, 2023), an example of which is shown in Fig. 2.

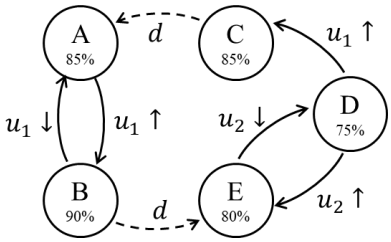

Figure 2. Example optimality graph with five modes A-E indicated by nodes, each with an associated product quality serving as KPI. Mode switches due to operator actions $\mathbf{u}$ and disturbances $\mathbf{d}$ are indicated by solid and dashed edges, respectively.

Figure 2 illustrates a hypothetical process with five modes identified from historical data and represented as nodes. The KPI for the process is product quality, indicated as a percentage associated with each mode. The graph is split into two groups separated by disturbances: modes A and B on the left and modes C, D and E on the right. While operator actions $\mathbf{u}$ have enabled shifting modes within groups (between A and B, or between C, D and E), there is no historical record of operator actions resulting in a shift from modes in one group to another. Modes in one group are therefore practically unreachable from modes in another group. As an example, the hypothetical process may have been operating in mode B when an unmeasured disturbance caused a mode shift to E, dropping quality from 90% to 80%. No operator actions enable a shift back to mode B. However, in the past a decrease in u_2 shifted the process from mode E to D, and a subsequent increase in u_1 shifted the process to mode C with an associated quality of 85%. The prescriptive analytics tools can thus recommend a set of operator actions that will maximize quality under the current process constraints. This paper aims to illustrate the development of an optimality graph to inform operator actions.

We explicitly require that the prescriptive analytics pipeline should be as straightforward as possible to encourage industrial adoption. To this end, we actively avoid YAMLA (Yet Another Machine Learning Algorithm) and consistently apply straightforward, established machine learning techniques that are readily available in well-supported libraries (e.g., Scikit-learn) for each step. This ensures the method is straightforward to implement (in fact, a carefully constructed prompt to a large language model (LLM), should be nearly sufficient), enabling rapid evaluation of a process dataset to assess whether the prescriptive analytics approach holds any promise before investing additional resources to develop a refined, process specific solution.

2. METHODOLOGY

2.1 Data wrangling and associated challenges

It is well known that data ingestion and preprocessing is often the most time-consuming component of a data analytics project. Unfortunately, this step is process-specific: no one-size-fits all approach is possible. We highlight three challenges that will be faced during the data wrangling stages of optimality graph development, and present no solutions.

The optimality graph method implicitly assumes that the process state $\mathbf{x}$ can be estimated from the online measurements $\mathbf{y}$ to subsequently infer the current operating mode $\mathcal{M}$, and that a KPI z can be associated with each mode thus identified, i.e., $z = z[\mathcal{M}] = z[g(\mathbf{x})] = z\left[g\left(h^{-1}(\mathbf{y})\right)\right]$. In other words, $\mathcal{M} = f(\mathbf{y})$ *by means of accurate state estimation* $\mathbf{x} = h^{-1}(\mathbf{y})$. It may be that the measurement space $\mathbf{y}$ is not sufficiently feature-rich, leading to a low ratio of between-cluster KPI variance to within-cluster KPI variance such that there is no real distinction between clusters. This may ultimately be an issue related to process observability (Auret, 2026) since the KPI is purely a function of the (sufficiently rich) state representation $\mathbf{x}$, in which case it must be addressed using instrumentation, not machine learning.

Secondly, even if there is sufficient observability, clusters may not be readily apparent without extensive feature engineering. This is particularly true for dynamically varying processes with long delays, where the measured values in one unit may only affect the process KPI much later. The measurements $\mathbf{y}(t)$ provide a snapshot of the process at a given moment t, but it may be necessary to combine current and past measurements to accurately estimate the current state $\mathbf{x}$. Time series alignment as well as dynamic embedding can be used to include lagged variables as additional features (see e.g. Qin et al., 2021), but this may result in a rapid increase in feature space dimensionality, thereby introducing a host of new issues.

Finally, a single process KPI may not be readily determined or assigned to a specific moment in time or a single process mode. Identifying a set of KPIs and retrospectively associating these with historic data may require in-depth process knowledge, interviews with plant personnel, etc. Multiple KPIs can be combined using suitable weightings or preference functions, or a multi-objective optimization strategy could be attempted. A single, readily assigned KPI is recommended for initial analysis: more resources may be invested in identifying and assigning additional KPIs of more direct relevance to plant performance.

We simply assume that the data wrangling process results in a set of n samples of q-dimensional engineered features $\mathbf{y}_k$ arranged in an array $Y \in \mathbb{R}^{n \times q}$, with an individual KPI value z_k associated with each sample, arranged in a vector $\mathbf{z} \in \mathbb{R}^n$. For simplicity, we retain the symbol $\mathbf{y}$ to represent engineered features obtained directly from online measurements.

2.2 Dimensionality reduction and clustering

Dimensionality reduction is often required prior to clustering to avoid the curse of dimensionality, particularly to limit the effect of highly correlated variables (often introduced by a naïve embedding approach). While dimensionality reduction is typically framed as an unsupervised learning task, our approach aims at identifying clusters in such a way that the KPI varies significantly from one cluster to the next (Meyer, 2023): in other words, we have access to a target variable. Many supervised learning techniques implicitly include dimensionality reduction: these range from simple linear methods such as Partial Least Squares (PLS) to complex approaches such as deep neural networks passing through bottleneck layers. The general approach is to project the features into a lower dimensional latent space $\mathbf{t} = \pi(\mathbf{y})$ with $\mathbf{t} \in \mathbb{R}^d$ and $d < q$, followed by regression of the target variable z onto the lower dimensional latent space, $z = g(\mathbf{t})$. In PLS, both the projection and regression functions are linear.

Avoiding YAMLA, we propose applying supervised regression techniques with implicit dimensionality reduction that are available in well-established libraries first, before developing any type of bespoke method. The simplicity of PLS makes it an ideal first candidate (Meyer, 2023).

Clustering can then be performed on the latent variables $\mathbf{t}$. We recommend *k*-means clustering as a first attempt. However, one should bear in mind that clustering occurs in the feature space and does not account for the time-series nature of data. A common problem is that consecutive data points may be assigned to different clusters on an alternating basis, resulting in rapid "switching" between clusters which is inconsistent with our understanding of process modes. This is illustrated in Fig. 3 for a dataset consisting of level and temperature measurements. Process intuition would indicate that clusters A and B should be combined into a single cluster to avoid the rapid mode switching. This is trivial to see in the fictitious example but becomes more difficult in higher dimensions with many observations and clusters. Other clustering methods have been developed to address similar challenges, but these are not typically implemented in common libraries. We avoid using YAMLA to address the issue and, instead of developing a bespoke clustering method, use a confusion matrix to detect rapid mode switching instead. By computing the confusion matrix where we use the cluster assignment at time t as the "true" class and the cluster assignment at time $t + 1$ as the "predicted" class, the confusion matrix will show how frequently an observation switches from one cluster at time t to another at $t + 1$. The confusion matrix for the above dataset is shown in Table 1.

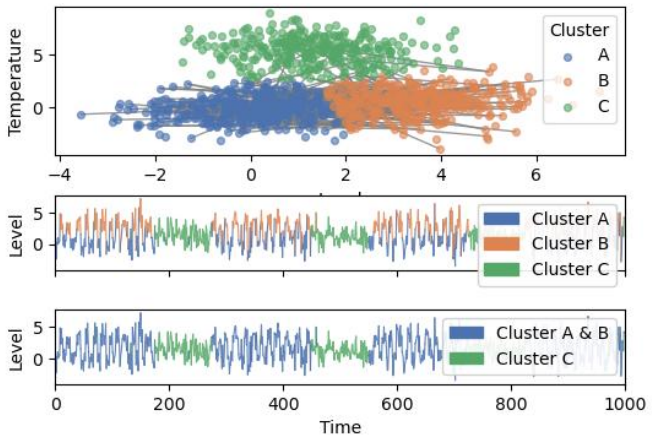

Figure 3. Time- series data in feature space (top) and as a time series plot (middle) colored according to cluster assignments. Gray lines indicate transitions between clusters from one timepoint to the next. Many transitions occur between clusters A and B, but few occur between A and C or B and C. Joining clusters A and B into a single cluster yields the modes shown in the bottom panel.

Table 1. Confusion matrix to assess cluster switching.

		$\mathcal{M}(t_{k+1})$		
		A	B	C
$\mathcal{M}(t_k)$	A	260	76	8
	B	76	269	7
	C	7	8	288

The very high number of switches between clusters A and B suggest that these should be combined into a single cluster, representing a single process mode. This approach does not alleviate all spurious mode switches, but it does greatly alleviate the problem. Further, it is well suited for use with *k*-means clustering: simply select a large initial value for *k*, then successively merge clusters with frequent switching to naturally reduce the number of clusters until a reasonable result is obtained.

The within- and between cluster KPI distribution is a useful metric to assess the efficacy of the dimensionality reduction and clustering steps. A simple approach is to consider the boxplots of KPIs grouped according to cluster, arranged in order of ascending average KPI. The boxplots reveal the extent to which the clustering effectively discriminates between process modes with distinct performance. A lack of significantly different KPIs between clusters may suggest adjusting the dimensionality reduction and/or clustering method hyperparameters: see the case study in Section 3.

2.3 Mode labelling

Clustering supplies non-descriptive labels, but intuitive labels are more useful: SHAP analysis (Lundberg et al., 2017) provides a means of developing descriptive mode labels by calculating the contribution of each feature to a supervised model output. Since clustering is an unsupervised learning task, SHAP cannot be applied directly. To circumvent this problem, a classifier is trained to predict the cluster assignments based on input data, and SHAP analysis is used to determine which features most influence the predicted probability for each class. The classifier will be useful later during online mode classification as well. Figure 4 shows SHAP values for each feature in the synthetic level-temperature dataset. The magnitude of the SHAP values indicate the strength of the contribution to the model output (in this case, probability of belonging to mode C). It is clear from this trivial data set that the combined mode A+B is associated with a low temperature and mode C with a high temperature, while level does not have a significant impact on the predictions. The modes can therefore be renamed "low temperature" and "high temperature", respectively. Mode labelling is also readily automated using LLMs.

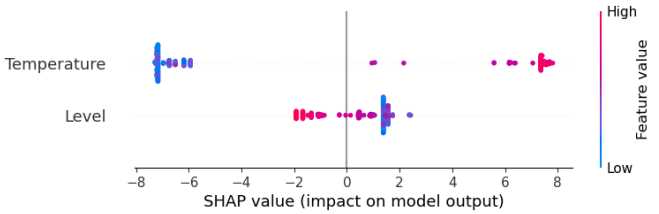

Figure 4. SHAP values indicating the contribution of each feature to the probability that an observation belongs to mode C, e.g. high temperature results in an increase in probability that an observation belongs to mode C.

2.4 Transitions

Once the data has been clustered and the operating modes identified and renamed, the transitions between modes must be identified. This is most easily accomplished by avoiding YAMLA and simply plotting the time series data for operator actions **u** with markers colored according to cluster assignment, as in the bottom panel of Fig. 3. Wherever a mode shift is indicated by a color change, the engineer identifies the corresponding change in input variable: if none can be discerned, assume the change is due to a disturbance.

An alternative is to calculate the variance of the inputs over a moving window and associate mode changes with inputs of high variance. This does not require YAMLA but a rather involved coding task ideally suited to an LLM. Finally, the mode labels developed in the previous step may also provide an indication of the actions responsible for a mode change, e.g., switching from a mode labelled "high temperature" to "low temperature" already indicates that a steam valve may be associated with the change.

2.5 Online prescriptive analytics

The final step involves visualizing the developed optimality graph displaying the process modes as nodes and transitions as edges and locating the current operating mode on the graph. The classifier developed during the cluster labelling step is ideally suited to the latter task. Once the current operating mode is known, the optimal reachable mode (i.e., maximal KPI mode that may be reached through operator actions) is identified and the necessary operator actions to facilitate the transition is prescribed.

3. CASE STUDY

The method was applied to a publicly available dataset consisting of 23 features describing an iron ore flotation process[1]. The percentage silica in the product was used as KPI

[1] https://www.kaggle.com/datasets/edumagalhaes/quality-prediction-in-a-mining-process

(a lower value is better) and data was downsampled to every 15 minutes resulting in ~16k observations; no further data wrangling or feature engineering was performed. PLS with *k*-means was used for dimensionality reduction and the described method was used to reduce the number of process modes to six. Figure 5 shows a boxplot of the KPI grouped by mode: the method effectively identifies modes with distinct KPI means and variabilities.

LightGBM was used as classifier to predict process modes and SHAP analysis was performed to generate mode labels. The SHAP values for the four features with greatest model impact (as measured by mean absolute SHAP value) for modes 1-3 are shown in Fig. 6 with the corresponding mode labels in Table 2, to illustrate the application.

Inspection of the time series data colored according to process mode enabled the identification of actions resulting in mode shifts for all changes, except for the mode shift from 3 to 5 which is subsequently classified as a disturbance. The resulting optimality graph is shown in Fig. 7. The graph shows that adjusting the air flow rate enabled movement between modes 3 and 5, but a disturbance shifted the process to a different operating region where movement due to operator actions is limited to modes 1, 2, 4 and 5. Here, increasing the air and decreasing amina feed flowrates resulted in improved KPIs.

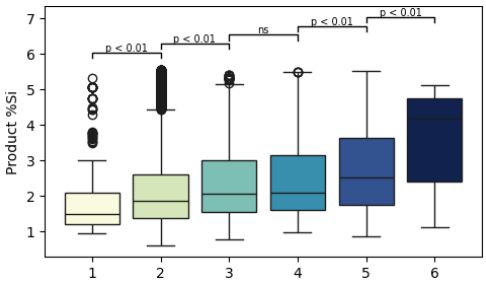

Figure 5. Boxplot of percentage silica in the product (KPI) grouped according to identified process modes, showing a gradual decrease in performance from mode 1 to 6.

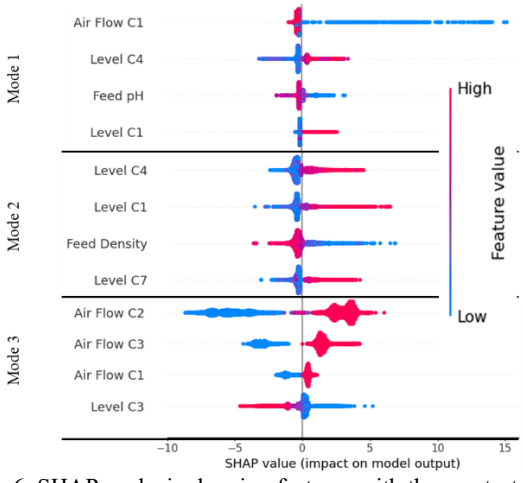

Figure 6. SHAP analysis showing features with the greatest impact on mode classification for the top three performing modes.

Table 2. Mode labels assigned using SHAP analysis

Mode	Label
1	Low air flow, low feed pH
2	Low feed density, high levels
3	High air flow, low levels

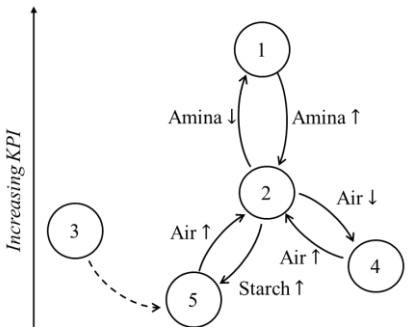

Figure 7. Optimality graph showing the transitions between identified modes due to operator actions (solid lines) and disturbances (dashed line).

4. CONCLUSIONS

The case study demonstrated a simple prescriptive analytics workflow based on readily available machine learning tools. No advanced machine learning methods are required: we rather emphasize the use of well-established methods in the hope that this will support widespread adoption.

The case study is by no means convincing in terms of performance improvement, as the between-mode KPI variation is limited. However, it does serve as a proof-of-concept for a relatively easy but powerful analysis on a moderately complex industrial dataset that may, with further feature engineering and method fine-tuning, result in significant economic benefit.

DECLARATION OF GENERATIVE AI AND AI-ASSISTED TECHNOLOGIES IN THE WRITING PROCESS

Claude Sonnet 4.6 was used to support coding and data analysis. After using this tool/service, the main author reviewed and edited the code as needed and takes full responsibility for the content of the publication.

REFERENCES

Auret, L. (2026). Machine Learning for Industrial Process Monitoring. *Encyclopedia of Systems and Control Engineering*, 24–52.

Liu, Y., Wang, F., Chang, Y., & Ma, R. (2015). Comprehensive economic index prediction based operating optimality assessment and nonoptimal cause identification for multimode processes. *Chemical Engineering Research and Design*, *97*, 77–90.

Lundberg, S. M., Allen, P. G., & Lee, S.-I. (2017). A Unified Approach to Interpreting Model Predictions. *Advances in Neural Information Processing Systems*, *30*.

Meyer, T. (2023). *Optimality assessment with optimality recovery for multi-modal process operations*. Stellenbosch : Stellenbosch University.

Qin, S. J., Liu, Y., & Dong, Y. (2021). Plant-wide troubleshooting and diagnosis using dynamic embedded latent feature analysis. *Computers & Chemical Engineering*, *152*(1)

Tan, S., Wang, F., Peng, J., Chang, Y., & Wang, S. (2011). Multimode Process Monitoring Based on Mode Identification. *Industrial and Engineering Chemistry Research*, *51*(1), 374–388.