

Data-Based Multi-Fidelity Modeling for Online Sensors Correction

Rastislav Fáber^{a,1}, Marco Vaccari^b, Riccardo Bacci di Capaci^b, Karol Ľubušký^c, Gabriele Pannocchia^b, Radoslav Paulen^a

^aFaculty of Chemical and Food Technology, Slovak University of Technology in Bratislava, Radlinského 9, 812 37 Bratislava, Slovakia

^bDepartment of Civil and Industrial Engineering, University of Pisa, Largo Lucio Lazzarino, 561 26 Pisa, Italy

^cSlovnaft, a.s., Vlčie hrdlo 1, 824 12 Bratislava, Slovakia

Abstract

Most plants within process industries employ frequent low-fidelity (LF) online sensor data together with sparse high-fidelity (HF) laboratory measurements, e.g., for product quality monitoring. While LF data are used for real-time operation, HF data recalibrate LF sensors occasionally. It is though rare that historical HF data are used for long-term improvement of LF sensors. We present a multi-fidelity (MF) soft-sensor framework that combines these two data sources. In two studied use cases, the proposed MF model reduces the prediction error by 20–50 % compared to LF sensors and reproduces HF trends with noticeable accuracy. The proposed method is general and transferable to other processes with similar data structure, providing interpretable results for improved monitoring and control.

Keywords: Multi-fidelity modeling, Soft sensors, Feature selection, Process monitoring, Industrial data analytics

^{*}rastislav.faber@stuba.sk (R. Fáber)

1. Introduction

Modern industrial plants collect large volumes of data from many sensors and stream analyzers that operate at different sampling rates and accuracy levels (Yin et al., 2015). Online instruments provide high-frequency information about the current plant state, yet they can suffer from drift, calibration issues, and measurement noise (Volpe and Wehr, 2016). Because of the accuracy issues, their records can be termed as low-fidelity (LF) data. Laboratory analyses, in contrast, are slower and more expensive, yet they provide traceable and reliable measurements, hence high-fidelity (HF) data. The imbalance between HF and LF data frequency and data quality hinders accurate real-time process monitoring.

In practice, when monitoring product quality, plant personnel treats laboratory measurements as a reference, while online analyzer data are used only if they remain close to the laboratory trend (Alauddin et al., 2023). Routine bias-correction or scaling procedures are often applied to align analyzer readings with laboratory values, but these data adjustments often arrive late, sometimes even days or weeks after the anomaly occurrence. A question arises whether, based on combined historical lab data (HF) and online analyzer data (LF), one can deploy a refinement mechanism that learns the corrective actions from historical records.

In spirit, prediction-correction estimation approaches, such as Kalman filter (Jiang et al., 2021), implement measurement-data refinement, where low-fidelity first-principles process model signal is repeatedly corrected by high-fidelity data. Even uncertainty estimates can be obtained in this approach, which makes it attractive. Yet, building and validating detailed

first-principles models for large-scale units is costly, and parameter estimation under plant uncertainty is often not feasible (Pantelides and Renfro, 2013). This motivates data-driven frameworks that can use LF and HF information without a full mechanistic description. In this context, multi-fidelity (MF) soft sensors have been proposed, starting from co-kriging schemes that link correlated LF and HF data (Forrester et al., 2007) and later extending to deep Gaussian process models for MF learning (Raissi and Karniadakis, 2016). Recent reviews summarize available MF architectures and their use in process optimization workflows (Peherstorfer et al., 2018; Fernández-Godino, 2023).

Generally, data-driven soft sensors, also called inferential models, estimate hard-to-measure quantities from correlated process variables (Kadlec et al., 2009). They use available plant measurements, such as temperatures, flow rates, and pressures, to predict variables whose sensing can exhibit unreliable behavior or can only be performed occasionally in the laboratory. Traditional soft sensors are designed on single-fidelity data, mostly on historical HF laboratory measurements (Curreri et al., 2020). Soft sensors trained on sparse HF data can, however, struggle to accurately capture process dynamics patterns (Fortuna et al., 2005). While it is technically feasible to train a soft sensor on LF data, the models based only on LF data risk propagating systematic bias and drift. This has to be borne in mind when designing an MF soft sensor.

Beside standard linear and non-linear regression, in the last decade, many works have applied machine-learning models to industrial soft sensor design. Recurrent and long short-term memory (LSTM) networks can be used to

represent nonlinear plant dynamics and temporal dependencies (Shah et al., 2020; Yuan et al., 2021). Other authors (Wang et al., 2019) further refine neural-network architectures and training procedures. While these methods can achieve low prediction errors, they require large amount of data for consistent training and are often difficult to interpret and to maintain in the long term.

To overcome these limitations, we design a MF soft sensor framework based on Gaussian Process Regression (GPR). We first construct an LF soft sensor using selected process variables. The discrepancy between LF and HF measurements is then modeled explicitly and used to correct the LF trajectory toward the HF reference.

The MF modeling itself is well established. Several aforementioned MF approaches have been proposed in the literature. The contribution of this work lies in a constrained and deployment-oriented MF workflow that combines three elements. First, a domain-knowledge-guided feature engineering step (Nargesian et al., 2017) filters input variables to retain only physically meaningful and maintainable sensors. Second, dynamic LF modeling captures transport delays and short-term process memory. Third, a structured multi-level cross-validation and kernel-selection strategy is implemented to avoid data leakage and ensure reproducible model selection.

Residual-based MF correction variants, denoted as Analyzer Correction (AC) and Predictor Correction (PC), together with well-understood linear and nonlinear single-fidelity regression models, were examined in earlier work (Fáber et al., 2025a,b), with emphasis on initial data analysis and proof-of-concept comparison of models with and without the correction structures

on limited datasets.

The present manuscript extends this preliminary work by embedding these structures into a unified formulation of static, dynamic, and MF models, and by introducing systematic model tuning, kernel selection, and multi-level cross-validation. The complete workflow is evaluated on two case studies: the Tennessee Eastman Process benchmark (TEP) and a full-scale industrial alkylation unit. This dual validation allows us to show that the proposed MF framework can deliver accurate and interpretable predictions for both, synthetic and real-world, datasets.

2. Problem Definition

Let us define a vector of process variables $\mathbf{x} \in \mathbb{R}^p$ and a process output variable $y \in \mathbb{R}$. The output is tied to the process variables by an unknown mapping:

$$y = F(\mathbf{x}). \quad (1)$$

The goal is to identify a surrogate mapping (model) \hat{F} so that

$$y = \hat{y} + \varepsilon, \quad \text{with } \hat{y} := \hat{F}(\mathbf{x}), \quad (2)$$

where \hat{y} is the model prediction and ε is the random model error. One can restate the model prediction from (2) as

$$\hat{y} = f(\boldsymbol{\phi}, \boldsymbol{\theta}), \quad \text{with } \boldsymbol{\phi} := \boldsymbol{\phi}(\mathbf{x}), \quad (3)$$

where $\boldsymbol{\phi}$ denotes a vector of regressors (features in machine-learning terminology) — (non)linear transformations and combinations of process variables — and $\boldsymbol{\theta}$ is the vector of model parameters.

For the purpose of modeling, historical data are available. Input matrix \mathbf{X} collects the (trusted) online sensor measurements. Let p denote the number of available input variables and let n denote the number of historical samples in \mathbf{X} . The matrix can be stated as

$$\mathbf{X} \in \mathbb{R}^{n \times p}, \quad \text{with } i^{\text{th}} \text{ row } \mathbf{x}_i^\top, \mathbf{x}_i \in \mathbb{R}^p, \quad (4)$$

where we use $^\top$ for the transpose. Vectors \mathbf{y}_{LF} and \mathbf{y}_{HF} contain the measurements of the same process output, sensed online (frequent but partially unreliable low-fidelity data) and by the laboratory analysis (sparse but accurate high-fidelity data), respectively:

$$\mathbf{y}_{\text{LF}} \in \mathbb{R}^{n_{\text{LF}}}, \quad \mathbf{y}_{\text{HF}} \in \mathbb{R}^{n_{\text{HF}}}, \quad (5)$$

where n_{LF} and n_{HF} are the numbers of LF and HF data with $n_{\text{HF}} \ll n_{\text{LF}} \leq n$. One can assume $n = n_{\text{LF}}$ for practical purposes. We define the LF timestamp set and LF measurement index set as

$$\mathcal{T}_{\text{LF}} := \{t_i\}_{i \in \mathcal{I}_{\text{LF}}}, \quad \mathcal{I}_{\text{LF}} := \{1, \dots, n_{\text{LF}}\}. \quad (6)$$

HF timestamps form a subset of LF timestamps. For the corresponding sets we define:

$$\mathcal{T}_{\text{HF}} := \{\tau_j\}_{j \in \mathcal{I}_{\text{HF}}} \subseteq \mathcal{T}_{\text{LF}}, \quad \mathcal{I}_{\text{HF}} := \{1, \dots, n_{\text{HF}}\}. \quad (7)$$

The use of data with certain fidelity level for model building and consequently the implied model fidelity can be denoted by the fidelity index f . The index values $f \in \{\text{LF}, \text{HF}\}$ denote a single-fidelity model built using LF or HF output measurements, respectively, and, $f = \text{MF}$ denotes a multi-fidelity model, where both LF and HF data were used mutually for model

building. Regarding temporal characteristic of the prediction model, we use $t \in \{\text{stat}, \text{dyn}\}$ to denote static or dynamic nature of the model, respectively.

At this point, several model configurations can be distinguished:

- Static linear model with the point prediction

$$\hat{y}_{f,i}^{\text{stat}} = f(\phi_{f,i}^{\text{stat}}, \boldsymbol{\theta}) := \boldsymbol{\theta}^\top \phi_{f,i}^{\text{stat}}, \quad \text{with } f \in \{\text{LF}, \text{HF}\}, \quad (8)$$

where, in the simplest case, $\phi_{f,i}^{\text{stat}}$ is formed by a subset of process variables $\tilde{\mathbf{x}}_i \in \mathbb{R}^r$ chosen from the columns \mathbf{x}_i^\top , i.e., $r \leq p$.

- Dynamic linear model with the point prediction $\hat{y}_{f,i}^{\text{dyn}}$ where the functional form of (8) is preserved but the regressor vector

$$\phi_{f,i}^{\text{dyn}} = [\tilde{\mathbf{x}}_i^\top, \tilde{\mathbf{x}}_{i-1}^\top, \tilde{\mathbf{x}}_{i-2}^\top, \dots, \tilde{\mathbf{x}}_{i-z}^\top, y_{f,i-1}, y_{f,i-2}, \dots, y_{f,i-l}]^\top, \quad (9)$$

in the simplest model form, contains lagged inputs and outputs, where z and l denote the input and output orders of the model, respectively.

- Nonlinear model

$$\hat{y}_{f,i}^t = f(\phi_{f,i}^t, \boldsymbol{\theta}), \quad \text{with } f \in \{\text{LF}, \text{HF}\}, t \in \{\text{stat}, \text{dyn}\} \quad (10)$$

- Multi-fidelity model can take any of the forms (8) or (10) and can be static or dynamic. Its main characteristic lies in a joint feature vector so that

$$\hat{y}_{\text{MF},i} = f(\phi_{\text{MF},i}, \boldsymbol{\theta}), \quad \text{with } \phi_{\text{MF},i} = [\phi_{\text{LF},i}^{t_{\text{LF}},\top}, \phi_{\text{HF},i}^{t_{\text{HF}},\top}]^\top, \quad (11)$$

where, by using $t_{\text{LF}}, t_{\text{HF}} \in \{\text{stat}, \text{dyn}\}$, we explicitly outline that the MF model can blend LF and HF data, e.g., dynamic LF features and

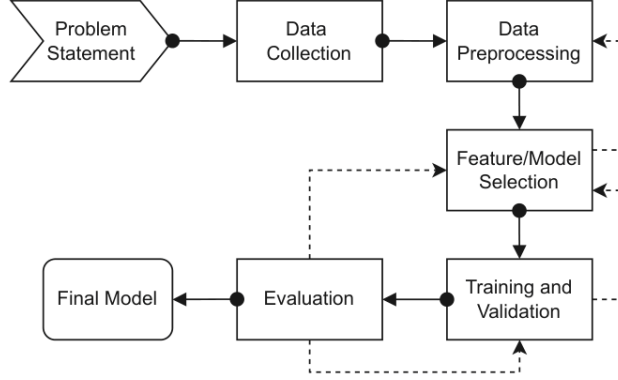


Figure 1: Data-based model development workflow.

static HF regressors. Simpler structures can be used such as $\phi_{\text{MF},i} = [\hat{y}_{\text{LF},i}^{\text{tLF}}, \phi_{\text{HF},i}^{\text{tHF},\top}]^\top$, which can help in model overfitting through features decorrelation and reduction of feature space.

3. Methodology

We devote the following section to present a practical approach to model development. The presentation (illustrated in Fig. 1) uses a typical procedure applied by an engineer in practice in an extensive way and with progressively increasing level of model complexity. Several steps can be omitted or only applied to a certain degree saturating the user satisfaction. Model complexity and tuning choices are constrained using structured cross-validation and information criteria. This ensures that structural decisions (feature dimension, model order, kernel type) are determined by objective validation metrics rather than manual tuning.

3.1. Data Preprocessing

Industrial data often contain noise, outliers, and missing values due to plant disturbances, start-ups or shutdowns. We address this issue in the data preprocessing step. Additionally, we standardize the cleaned datasets.

We treat the input \mathbf{X} and output $\mathbf{y}_{\text{LF}}, \mathbf{y}_{\text{HF}}$ datasets by removing missing entries, constant-valued signals, and evident outliers with respect to the historical operating distribution. Outlying measurements can be detected by the Minimum Covariance Determinant (MCD) method (Rousseeuw and Van Driessen, 1999). For each point \mathbf{x}_i , the Mahalanobis distance can be calculated as

$$d_i = \sqrt{(\mathbf{x}_i - \boldsymbol{\mu})^\top \mathbf{S}^{-1} (\mathbf{x}_i - \boldsymbol{\mu})}, \quad (12)$$

where $\boldsymbol{\mu}$ is the vector of means (column-wise mean of \mathbf{X}) and \mathbf{S} is the covariance matrix of \mathbf{X} . A point is statistically an outlier if $d_i^2 > \chi_{p,\alpha}^2$, where $\chi_{p,\alpha}^2$ is the $\alpha \times 100\%$ quantile of the chi-square distribution with p degrees of freedom. Removed points are substituted by linear interpolation before we standardize the dataset. This simple preprocessing can be applied to stream data once the prediction model is deployed to the plant. Outlier thresholds are fixed using the chi-square criterion and are not tuned to improve prediction performance.

3.2. Cross-Validation Data Split

To avoid model overfitting and improve model reliability, we perform cross-validation (CV) at various instances of training. Because the proposed framework involves feature selection, single-fidelity model design, and subsequent multi-fidelity residual correction, a single train-validation split is not

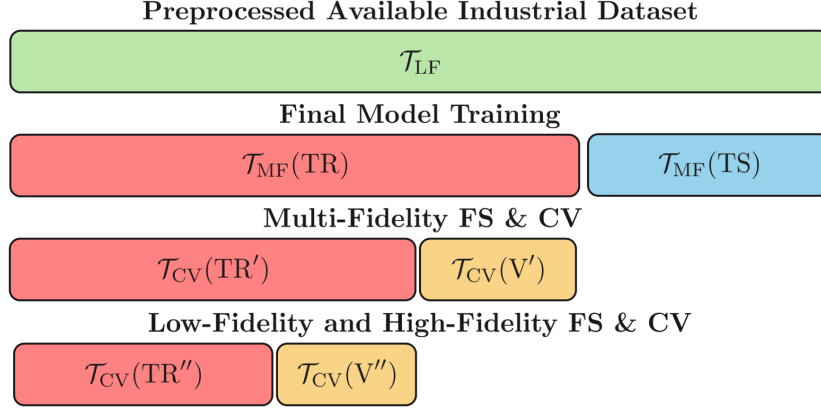


Figure 2: Comprehensive dataset partitioning illustrating the three-level data split.

sufficient. Separate data partitions are required to avoid information leakage between feature selection, LF model tuning, MF kernel selection, and final performance evaluation. Each split level isolates one modeling stage so that validation results remain unbiased and reproducible.

As we shall see later, training of an MF model might require application of CV at more than one level. For this purpose, we split the available cleaned data as sketched in Fig.2.

In the first level of splitting, we reserve a portion of data for the final single- and multi-fidelity model testing $\mathcal{T}_{MF}(TS)$ and use the remaining measurements for training of the MF model $\mathcal{T}_{MF}(TR)$. The left-out, “Final Model Training”, testing subset $\mathcal{T}_{MF}(TS)$ remains completely untouched during FS, single-fidelity model tuning, and MF kernel selection. All model-structure decisions are therefore taken only on inner splits, ensuring unbiased final evaluation. Formally,

$$\mathcal{T}_{LF} = \mathcal{T}_{MF}(TR) \cup \mathcal{T}_{MF}(TS), \quad \mathcal{T}_{MF}(TR) \cap \mathcal{T}_{MF}(TS) = \emptyset. \quad (13)$$

The second level of splitting corresponds to multi-fidelity CV. The MF training set $\mathcal{T}_{\text{MF}}(\text{TR})$ is divided into an MF training subset and an MF validation subset, used for MF model selection as

$$\mathcal{T}_{\text{MF}}(\text{TR}) = \mathcal{T}_{\text{CV}}(\text{TR}') \cup \mathcal{T}_{\text{CV}}(\text{V}'), \quad \mathcal{T}_{\text{CV}}(\text{TR}') \cap \mathcal{T}_{\text{CV}}(\text{V}') = \emptyset. \quad (14)$$

The third, and final, data split involves feature selection and CV-based design of single-fidelity models. Here, the subset $\mathcal{T}_{\text{CV}}(\text{TR}')$, is further split into single-fidelity training $\mathcal{T}_{\text{CV}}(\text{TR}'')$ and validation $\mathcal{T}_{\text{CV}}(\text{V}'')$ subsets. The corresponding timestamp sets follow:

$$\mathcal{T}_{\text{CV}}(\text{TR}') = \mathcal{T}_{\text{CV}}(\text{TR}'') \cup \mathcal{T}_{\text{CV}}(\text{V}''), \text{ with } \mathcal{T}_{\text{CV}}(\text{TR}'') \cap \mathcal{T}_{\text{CV}}(\text{V}'') = \emptyset. \quad (15)$$

Complementary to the timestamp sets, index sets $\mathcal{I}_{\text{MF}}(\text{TR})$, $\mathcal{I}_{\text{MF}}(\text{TS})$, $\mathcal{I}_{\text{CV}}(\text{TR}')$, $\mathcal{I}_{\text{CV}}(\text{V}')$, $\mathcal{I}_{\text{CV}}(\text{TR}'')$, and $\mathcal{I}_{\text{CV}}(\text{V}'')$ are created accordingly. When splitting the datasets, we ensure that, because of $\mathcal{T}_{\text{HF}} \subseteq \mathcal{T}_{\text{LF}}$, both $\mathcal{T}_{\text{MF}}(\text{TR})$ and $\mathcal{T}_{\text{MF}}(\text{TS})$ involve sufficient amounts of HF data for representative coverage.

3.3. Feature Selection

To improve model quality, we select a small subset of inputs that explain the output while providing accurate predictions across different operation points and are easy to audit and maintain. We use the following FS methods on $\mathcal{T}_{\text{CV}}(\text{TR}'') \cup \mathcal{T}_{\text{CV}}(\text{V}'')$. We use the expanding-window CV scheme on the LF data to capture temporal dependencies, and a standard k -fold CV on the HF data to aim for a robust subset of inputs for the comparably limited number of HF samples. This systematic approach allows us to select the best single-fidelity candidate \hat{y}_{f} as an additional input for the multi-fidelity

model. To assess the stability of the selection, we evaluate several standard FS methods on all inputs \mathbf{X} against \mathbf{y}_{LF} or \mathbf{y}_{HF} on $\mathcal{T}_{\text{MF}}(\text{TR})$. In practical deployment, DK post-selection can be applied directly, while the additional FS methods are included here mainly as comparative baselines.

Principal Component Regression (PCR): We derive a new subspace of principal components (PCs) by reducing the dimensionality of the preprocessed inputs \mathbf{X} . The output variable (either \mathbf{y}_{LF} , or \mathbf{y}_{HF}) is then regressed on the most relevant components to minimize the prediction error. For example, the feature vector of a linear static model reads $\phi_{\text{f},i}^{\text{stat, PCR}} = \tilde{\mathbf{z}}_i$, where $\tilde{\mathbf{z}}_i$ contains the first r right-singular vectors of \mathbf{X} from its singular value decomposition. The stopping criterion is based on the cumulative explained variance (typically 90 % EV), which determines the optimal number of PCs.

Partial Least Squares (PLS): We similarly compute a subspace of PCs that maximizes input-output covariance on the preprocessed dataset (Geladi and Kowalski, 1986). The output variable (either \mathbf{y}_{LF} , or \mathbf{y}_{HF}) is regressed on these latent variables. Inputs are, similar to PCA, ranked by the absolute weights that link the original columns of \mathbf{X} to the selected latent variables that minimize the prediction error. We can define

$$\mathbf{X} = \mathbf{T}\mathbf{P}^\top + \mathbf{E}, \quad \mathbf{y} = \mathbf{T}\mathbf{q} + \mathbf{r}, \quad (16)$$

where \mathbf{T} is the latent score matrix, \mathbf{P} and \mathbf{q} denote the loadings, and \mathbf{E} and \mathbf{r} the error terms. The feature vector of a linear static model reads as $\phi_{\text{f},i}^{\text{stat, PLS}} = \tilde{\mathbf{p}}_i$, where $\tilde{\mathbf{p}}_i$ contains the first r columns of \mathbf{P}^\top . The stopping criterion follows the same principle as in PCA, using cumulative EV, selecting the minimal number of PCs that capture sufficient covariance with the output (e.g., 90 %).

LASSO Regression (LASSO): Unlike the aforementioned PCR/PLS, which form a subspace of ranked latent components, LASSO performs direct variable selection by shrinking model parameters towards zero with an ℓ_1 penalty. One can solve

$$\min_{\boldsymbol{\theta}} \sum_{i \in \mathcal{I}_f} (y_{f,i}^t - f(\boldsymbol{\phi}_{f,i}^t, \boldsymbol{\theta}))^2 + \lambda \|\boldsymbol{\theta}_f\|_1. \quad (17)$$

Here λ controls the model sparsity. Relevant features are retained only if their addition reduces the prediction error by more than a defined threshold (e.g., $\delta = 0.005$), which serves as the stopping criterion. Near-zero parameters are discarded automatically (Santosa and Symes, 1986). In this work, LASSO is used for sparse regressor selection, while model evaluation remains based on squared-error metrics to ensure consistency with the remaining modeling approaches.

Stepwise Regression (SR): We build a subset of regressors by finding the most relevant inputs, typically using F-tests or t-tests, to explain the output variable. In a forward variant of SR, we start from no inputs and add, at each step, a single variable (column) from X that yields the largest drop in validation error. A backward variant can be applied, where we start from the full set of X and remove a single variable that gives the smallest increase in validation error. At each step, we refit the model on a training subset and accept a variable only if prediction error decreases. We define a stopping criterion when no single addition or removal of a variable improves the prediction by a defined threshold (e.g., $\delta = 0.005$).

The prediction error is assessed via Root Mean Squared Error (RMSE) on any subset $s \in \{\text{TR}, \text{V}, \text{TS}, \text{TR}', \dots\}$ with fidelity f . Formally, for any

trained model ξ of the form (2), we can define

$$\text{RMSE}_{\text{f}}^{\text{s}}(\xi) = \sqrt{\frac{1}{\text{card}(\mathcal{I}_{\text{f}}^{\text{s}})} \sum_{i \in \mathcal{I}_{\text{f}}^{\text{s}}} (y_{\text{f},i} - \xi_i)^2}. \quad (18)$$

To prevent overfitting, we use validation-to-training RMSE ratio as a complementary metric. A large ratio indicates that the model fits the training data well but does not generalize, while a ratio close to one indicates consistent predictive performance across training and validation data.

All input and output variables are standardized prior to model training. Consequently, RMSE values reported throughout this study are computed on standardized signals, ensuring scale consistency within each case study. For interpretability, percentage improvements are reported relative to the LF online analyzer baseline. In addition to RMSE, we evaluated Mean Absolute Error (MAE) as a complementary metric for the results assessment. The relative ranking of the models remained unchanged under MAE and it is thus not reported here.

Domain Knowledge (DK): After statistical ranking by the FS methods described above, the suggested variables are verified and post-selected using criteria reflecting feasibility and reliability in industrial deployment. The candidate regressors are prioritized in the order: process variables (PV) > controller outputs (OP) > setpoints (SP).

Measured PVs represent the physical state of the unit and are typically the most reliable and transferable signals across operating modes, controller tuning changes, and operator interventions. In contrast, OPs reflect the control policy (e.g., gain scheduling, saturation, anti-windup, mode changes, and manual overrides) and may therefore encode plant-specific control actions

rather than true process behavior. OP signals can be informative, but they are more sensitive to configuration changes and can exhibit implementation-specific behavior when controllers switch modes or when actuators saturate. SPs are supervisory targets and are frequently constant over long periods; consequently, they often provide limited excitation and can act as weak regressors unless they contain sustained temporal variation.

Variables failing to meet these criteria are replaced iteratively with the next-best-ranked candidates, and statistical criteria (e.g., RMSE) are recomputed after each adjustment. The selected regressor is denoted as $\phi_{f,i}^{\text{t, DK}}$.

3.4. Model Training

Once the features are selected, a model is trained via solving the least-squares problem

$$\min_{\theta} \sum_{i \in \mathcal{I}_f^s} (y_{f,i} - \hat{y}_{f,i}^{\text{t}})^2 \equiv \min_{\theta} \sum_{i \in \mathcal{I}_f^s} (y_{f,i} - f_f(\phi_{f,i}^{\text{t}}, \theta))^2. \quad (19)$$

When training a dynamic model (Eq. (8) with features like (9)), we use the well-established Information Criteria (IC) to tune the model order: Akaike Information Criterion (AIC), the small-sample corrected Akaike Information Criterion (AICc), and the Bayesian Information Criterion (BIC). For the appropriate model order, one selects the model with the lowest value of the chosen IC to limit its complexity (Efroymson, 1960). The dynamic order is selected by minimizing the chosen IC and is not manually tuned.

Complementary to the linear models (8), we use two types of nonlinear prediction models.

Gaussian Process Regression (GPR). The GP prediction model is established as

$$f_{\text{f}}(\phi_{\text{f},i}, \boldsymbol{\theta}) \sim \mathcal{GP}(\mu(\phi_{\text{f},i}), k(\phi_{\text{f},i}, \phi_{\text{f},j})), \quad (20)$$

with the mean $\mu(\cdot)$ used as $\hat{y}_{\text{f},i}$ and a covariance kernel $k(\cdot, \cdot)$ with hyperparameters $\boldsymbol{\theta}$.

We select the GP kernel structure using validation performance within the defined CV split, to avoid manual kernel tuning, and provides the lowest overall prediction error. We study the following (composite) kernels:

- k_1 : *Radial Basis Function (RBF)* - captures general process trends.

The signal variance σ^2 defines the average distance of the function from its mean, and lengthscale ℓ_{RBF} defines the smoothness of the function:

$$k_1(\phi_i, \phi_j) = \sigma^2 \exp\left(-\frac{\|\phi_i - \phi_j\|^2}{2\ell_{\text{RBF}}^2}\right). \quad (21)$$

- k_2 : *Scaled RBF* - product of a scaling constant C and an RBF kernel, accounting for variability as $k_2(\phi_i, \phi_j) = C \times k_1(\phi_i, \phi_j)$.

- k_3 : *RBF + White Noise* - combines smooth trends with a noise term to represent measurement uncertainty as $k_3(\phi_i, \phi_j) = k_1(\phi_i, \phi_j) + \sigma_{\text{n}}^2 \delta_{ij}$, where the standard deviation of the noise is σ_{n}^2 and δ_{ij} is the Kronecker delta (1 if $i = j$, 0 otherwise).

- k_4 : *Scaled RBF * Periodic + White Noise* - captures both smooth and cyclic behavior under noisy conditions. Here d defines the distance between peaks of the function and ℓ_{PER} its lengthscale in the same way as in the RBF kernel:

$$k_4(\phi_i, \phi_j) = k_2(\phi_i, \phi_j) \times \sigma^2 \exp\left(-\frac{2 \sin^2\left(\frac{\pi}{d} \|\phi_i - \phi_j\|\right)}{\ell_{\text{PER}}^2}\right) + \sigma_{\text{n}}^2 \delta_{ij} \quad (22)$$

- k_5 : *Scaled RBF + Periodic + White Noise* - an additive version combining trend, periodicity, and noise effects:

$$k_5(\phi_i, \phi_j) = k_2(\phi_i, \phi_j) + \sigma^2 \exp\left(-\frac{2 \sin^2\left(\frac{\pi}{d} \|\phi_i - \phi_j\|\right)}{\ell_{\text{PER}}^2}\right) + \sigma_n^2 \delta_{ij} \quad (23)$$

The additive combination of kernels (sum) acts as an OR operator, while multiplication acts as an AND operator across input dimensions (Duvenaud, 2014).

Feed-Forward Neural Network (NN) and Autoencoder (AE). Both models use the neural-network architecture. An L -layer feed-forward network maps the input vector $\phi_{f,i}$ to the output $\hat{y}_{f,i}$ as

$$\begin{aligned} \mathbf{h}^{(1)} &= \sigma^{(1)}(\mathbf{W}^{(1)} \phi_{f,i}^t + \mathbf{b}^{(1)}), \dots, \mathbf{h}^{(i)} = \sigma^{(i)}(\mathbf{W}^{(i)} \mathbf{h}^{(i-1)} + \mathbf{b}^{(i)}), \dots, \\ \hat{y}_{f,i}^t &= \sigma^{(L)}(\mathbf{w}^{(L,T)} \mathbf{h}^{(L-1)} + b^{(L)}), \end{aligned} \quad (24)$$

where $\mathbf{W}^{(\cdot)}$ ($\mathbf{w}^{(\cdot)}$) and $\mathbf{b}^{(\cdot)}$ ($b^{(\cdot)}$) represent the layer-specific weight matrix (layer L weight vector) and bias vector (layer L bias), respectively, and $\sigma^{(i)}(\cdot)$ is an element-wise activation function (e.g., ReLU, SELU, tanh) that introduces the requisite nonlinearity.

The AE model can be seen as a nonlinear variant of PCA, where it first encodes the features

$$\begin{aligned} \mathbf{h}^{(1)} &= \sigma^{(1)}(\mathbf{W}^{(1)} \phi_{f,i}^t + \mathbf{b}^{(1)}), \dots, \mathbf{h}^{(i)} = \sigma^{(i)}(\mathbf{W}^{(i)} \mathbf{h}^{(i-1)} + \mathbf{b}^{(i)}), \dots, \\ \text{Enc}(\phi_{f,i}^t) &= \sigma^{(L)}(\mathbf{W}^{(L)} \mathbf{h}^{(L-1)} + \mathbf{b}^{(L)}). \end{aligned} \quad (25)$$

A regression (usually a linear one) is then performed on

$$\hat{y}_{f,i}^t = \boldsymbol{\theta}^\top \text{Enc}(\phi_{f,i}^t), \quad (26)$$

and the features are decoded

$$\begin{aligned} \tilde{\mathbf{h}}^{(1)} &= \tilde{\sigma}^{(1)} \left(\tilde{\mathbf{W}}^{(1)} \text{Enc}(\phi_{\text{f},i}^{\text{t}}) + \tilde{\mathbf{b}}^{(1)} \right), \dots, \tilde{\mathbf{h}}^{(i)} = \tilde{\sigma}^{(i)} \left(\tilde{\mathbf{W}}^{(i)} \tilde{\mathbf{h}}^{(i-1)} + \tilde{\mathbf{b}}^{(i)} \right), \dots, \\ \text{Dec}(\phi_{\text{f},i}^{\text{t}}) &= \tilde{\sigma}^{(L)} \left(\tilde{\mathbf{W}}^{(L)} \tilde{\mathbf{h}}^{(L-1)} + \tilde{\mathbf{b}}^{(L)} \right), \end{aligned} \quad (27)$$

where AE is trained so that the decoded features $\text{Dec}(\phi_{\text{f},i}^{\text{t}})$ match the original features $\phi_{\text{f},i}^{\text{t}}$ by minimizing the reconstruction loss.

Multi-fidelity modeling. Our aim is to train an MF model that is transparent, feasible for industrial implementation, and reliable when only a small number of HF samples are available, given the structural complexity of the general MF formulation (11). For this reason, we use two MF correction structures, Analyzer Correction (AC) and Predictor Correction (PC), which were introduced in our previous work (Fáber et al., 2025a,b). In this study, these structures are integrated into the complete workflow, including feature selection, dynamic LF modeling, and structured cross-validation for kernel and model selection.

(a) *Analyzer Correction (AC).* The idea behind the model is to provide a correction to the analyzer output values $y_{\text{LF},i}$ by learning the historical discrepancy

$$\Delta_i = y_{\text{HF},i} - y_{\text{LF},i}, \quad i \in \mathcal{I}_{\text{HF}}, \quad (28)$$

and form a corrected trajectory

$$\hat{y}_{\text{MF},i}^{\Delta\text{AC}} = y_{\text{LF},i} + \hat{\Delta}_i^{\text{AC}}, \quad i \in \mathcal{I}_{\text{LF}}, \quad (29)$$

where $\hat{\Delta}_i^{\text{AC}}$ is an MF model (11) trained on the residuals (28).

(b) *Predictor Correction (PC).* A similar idea is used, yet here an LF

predictor is built first to provide filtering for LF data.

$$\hat{y}_{\text{MF},i}^{\Delta_{\text{PC}}} = \hat{y}_{\text{LF},i} + \hat{\Delta}_i^{\text{PC}}, \quad i \in \mathcal{I}_{\text{LF}}, \quad (30)$$

where $\hat{\Delta}_i^{\text{PC}}$ is an MF model (11) trained on the residuals $\Delta_i = y_{\text{HF},i} - \hat{y}_{\text{LF},i}, \forall i \in \mathcal{I}_{\text{HF}}$.

In both variants (a) and (b), as well as in Eq. (11), we model f_{MF} using GPR. Our choice is motivated by the sparsity of HF measurements. GP models provide consistent training without large calibration datasets and require minimal model specification.

Several FS and single-fidelity model variants are evaluated as comparative baselines. The final recommended configuration is selected using structured cross-validation and information criteria, leading to a compact workflow with clearly defined and constrained tuning steps.

The proposed MF correction can be interpreted as a data-driven extension of classical analyzer bias updates used in advanced control applications. Unlike filtered offset corrections that are typically implemented within a process model, the proposed approach learns the discrepancy between LF and HF measurements directly from historical data and can be applied without an explicit first-principles representation of the process.

4. Implementation

We implement the methodology from Section 3 in Python. For data preprocessing step, the data are treated using the MCD method with $\alpha = 0.975$. The used TEP dataset (Averkiev, 2018) (1st case study) does not contain anomalies, so this step is not necessary therein.

Procedure 1 General workflow for model training and evaluation

Input: input matrix \mathbf{X} ;
 LF output \mathbf{y}_{LF} on \mathcal{T}_{LF} with index set \mathcal{I}_{LF} ;
 HF output \mathbf{y}_{HF} on $\mathcal{T}_{\text{HF}} \subset \mathcal{T}_{\text{LF}}$ with index set \mathcal{I}_{HF} .

1. Preprocess data.

- Remove invalid samples and evident outliers. Standardize the dataset as described in Sec. 3.1.
- Apply the hierarchical data split defined in Sec. 3.2 and illustrated in Fig. 2.

2. Select regressors.

- Apply the FS methods from Sec. 3.3 to columns of \mathbf{X} using $\mathcal{T}_{\text{CV}}(\text{TR}'')$.
- Train linear single-fidelity predictors for the candidate subsets.
- Select the subset with the lowest prediction error on $\mathcal{T}_{\text{CV}}(\mathbf{V}'')$.

3. Train single-fidelity models.

- Train candidate single-fidelity predictors \hat{y}_t^f with fidelity index $f \in \{\text{LF}, \text{HF}\}$ and temporal index $t \in \{\text{stat}, \text{dyn}\}$ as defined in Sec. 3.4 on $\mathcal{T}_{\text{CV}}(\text{TR}')$.
- Select the best-performing single-fidelity predictor using $\mathcal{T}_{\text{CV}}(\mathbf{V}')$.
- Select the GP kernel structure (standard or composite) for the MF model using $\mathcal{T}_{\text{CV}}(\text{TR}')$ and $\mathcal{T}_{\text{CV}}(\mathbf{V}')$ as described in Sec. 3.4.

4. Train multi-fidelity models.

- Use the selected single-fidelity predictor as the additional input, as defined in Eq. (11).
- Train the MF correction structures (AC and PC) on $\mathcal{T}_{\text{MF}}(\text{TR})$.

5. Evaluate candidate models.

- Evaluate all trained models on $\mathcal{T}_{\text{MF}}(\text{TS})$.
- Compute the prediction error using reference values at $i \in \mathcal{I}_{\text{HF}}$.
- Select the best-performing predictor \hat{y}^* .

Output: predictor \hat{y}^* with the best performance against HF data.

Data split for FS is done by running the expanding-window and k -fold CV for LF and HF models, respectively, over $\mathcal{T}_{\text{MF}}(\text{TR})$. A representative split is selected to train ($\mathcal{T}_{\text{CV}}(\text{TR}'')$) and validate ($\mathcal{T}_{\text{CV}}(\text{V}'')$) the models for FS of HF and LF models and the rest of the data at time points $\mathcal{T}_{\text{CV}}(\text{V}')$ is used for FS of MF models validation, while $\mathcal{T}_{\text{CV}}(\text{TR}') = \mathcal{T}_{\text{CV}}(\text{TR}'') \cup \mathcal{T}_{\text{CV}}(\text{V}'')$ is used for MF models initial training. The overall workflow for feature selection, model training, and final evaluation is summarized in Procedure 1.

Static linear regression and dimensionality-reduction methods (PCA, PLS) as well as GPR are implemented using the `scikit-learn` library (Pedregosa et al., 2011). NN learning is done via Keras (Chollet et al., 2015). System identification required for dynamic modeling is carried out using the `SIPPY` toolbox (Armenise et al., 2018). We opt for PARSIM-K method, which proved effective in our prior works.

Nonlinear single-fidelity models, both GP and feed-forward NN are trained using the DK features of the best-performing linear model. This keeps the input dimension comparable among the models for fair assessment. The AE model uses all available inputs but compresses the data to a latent dimension equal to the size of the DK subset. The GP model kernel functions are tuned via CV. The feed-forward NN involves three hidden layers with 20, 10, and 6 neurons, using SELU, tanh, and SELU activations, respectively, and a single linear output neuron. The AE involves, comparably to the NN structure, two hidden layers with 20 and 10 neurons in the encoder layers, where SELU and tanh activation functions are used in the hidden layers, with a linear activation at the output layer.

The MF model is implemented as GP with features $\phi_{\text{MF},i} = [\hat{y}_{\text{LF},i}^{\text{tLF}}, \phi_{\text{HF},i}^{\text{tHF},\text{T}}]^{\text{T}}$.

Procedure 2 Proposed MF workflow: predictor correction (PC)

Input: preprocessed DK subset of columns of \mathbf{X} ;
 LF output \mathbf{y}_{LF} on \mathcal{T}_{LF} with index set \mathcal{I}_{LF} ;
 HF output \mathbf{y}_{HF} on $\mathcal{T}_{\text{HF}} \subset \mathcal{T}_{\text{LF}}$ with index set \mathcal{I}_{HF} .

1. **Train single-fidelity predictor.**

- Identify the dynamic single-fidelity predictor $\hat{y}_{\text{LF}}^{\text{dyn}}$ on the preprocessed DK inputs.
- Select the model order using BIC.

2. **Train MF correction model (PC).**

- Add the dynamic single-fidelity prediction $\hat{y}_{\text{LF}}^{\text{dyn}}$ as an additional feature to the DK input subset.
- Use the *Scaled RBF* kernel (k_2) for the MF GP model.
- Train the GPR model to predict the offset $\Delta_i = y_{\text{HF},i} - \hat{y}_{\text{LF},i}^{\text{dyn}}$.
- Correct the dynamic single-fidelity prediction using the GP-predicted offset $\hat{\Delta}$ to obtain the PC MF prediction.

Output: trained predictor-correction MF model $\hat{y}_{\text{MF}}^{\Delta\text{PC}}$.

We select the best performing LF model, as the additional input into the MF model, based on the RMSE computed against the HF data. We perform MF model CV for kernel tuning using the training subset $\mathcal{T}_{\text{CV}}(\text{TR}')$, which maximizes the number of available LF–HF pairs for kernel tuning. This way we improve the conditioning of the covariance matrix and reduce the variance of the estimated hyperparameters. Kernel selection is based on the median RMSE and its variance computed over repeated GP training runs, where each run corresponds to an independent MF model training over $\mathcal{T}_{\text{CV}}(\text{V}')$. After kernel selection, the final MF model is retrained on the set $\mathcal{T}_{\text{MF}}(\text{TR})$. The proposed implementation of the predictor-correction MF model is summarized in Procedure 2.

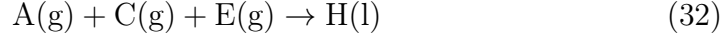
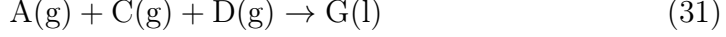
5. Case Study 1: Tennessee Eastman Process (TEP)

We first demonstrate the proposed framework on the TEP, a standard benchmark with coupled reaction-separation-recycle dynamics (Reinartz et al., 2021).

5.1. Process Overview

The TEP flowsheet consists of a mixing section, reactor, condenser, separator, stripper, product purification unit, purge stream, and a recycle loop. Fig. 3 shows the main process units and recycle streams. The plant produces two liquid products (G, H) from four gaseous reactants (A, C, D, E) with by-product F. Main irreversible, exothermic reactions are described as (Russell

et al., 2000):



5.2. Dataset and Signals

The dataset contains 52 variables, comprising 41 measured process variables and 11 manipulated inputs with $n_{\text{LF}} = 1,000$ data points. The objective is to study product quality related behavior to ensure efficient material utilization and prevent unnecessary accumulation or recycling of reactants. We select the output variable as the concentration of component C (y_{LF}) in the purge stream (Analyzer A1 in Fig. 3). This output is directly influenced by reactor conversion and separation efficiency. The depicted red markers (DK₁–DK₄) indicate sensor locations corresponding to key process variables.

Since laboratory measurements are not available in the dataset, we construct a pseudo HF signal y_{HF} to emulate sparse laboratory analyses. The objective is to approximate laboratory-type measurements that reflect the dominant process trend while being less sensitive to short-term analyzer fluctuations. For this purpose, the LF signal y_{LF} is smoothed using a 20-sample moving average with a fixed index offset $b = 5$:

$$y_{\text{HF}}(i) = \frac{1}{20} \sum_{k=0}^{19} y_{\text{LF}}(i - k + b). \quad (35)$$

The moving-average operation suppresses high-frequency analyzer noise so that the signal captures trend-level information. The index offset prevents

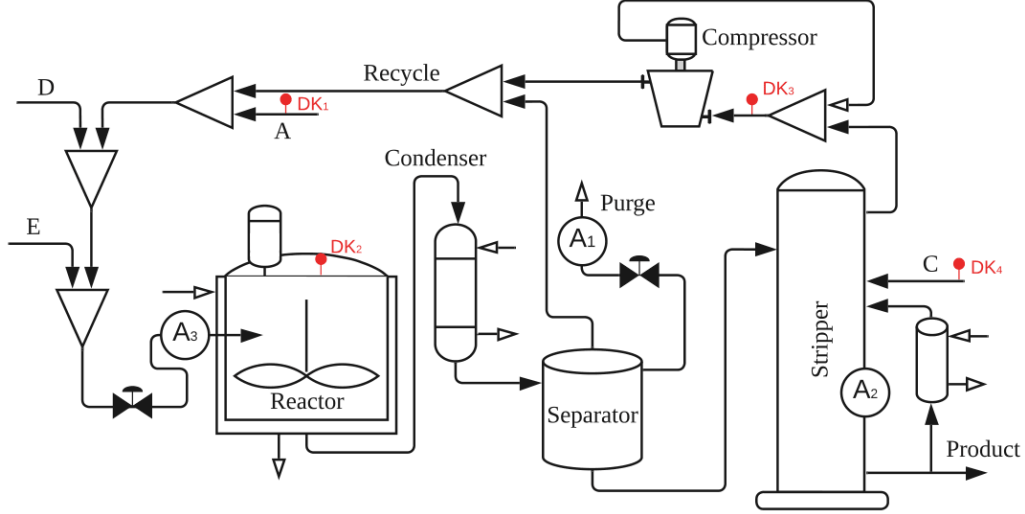


Figure 3: Simplified schematic of the Tennessee Eastman process, showing key units, recycle streams, and the placement of critical sensors.

direct alignment with LF samples and avoids overlap between LF and HF signals. The resulting signal is downsampled to $n_{\text{HF}} = 28$ evenly spaced samples to mimic laboratory-like sparsity.

We split the dataset chronologically: 70 % into $\mathcal{T}_{\text{MF}}(\text{TR})$ and 30 % into $\mathcal{T}_{\text{MF}}(\text{TS})$. This yields 700 LF samples and 18 HF samples in $\mathcal{T}_{\text{MF}}(\text{TR})$, and 300 LF samples and 10 HF samples in $\mathcal{T}_{\text{MF}}(\text{TS})$.

5.3. Results

The results are presented in sequence, starting from CV and FS, followed by single-fidelity model comparison (linear, static, dynamic, and nonlinear), to final model assessment.

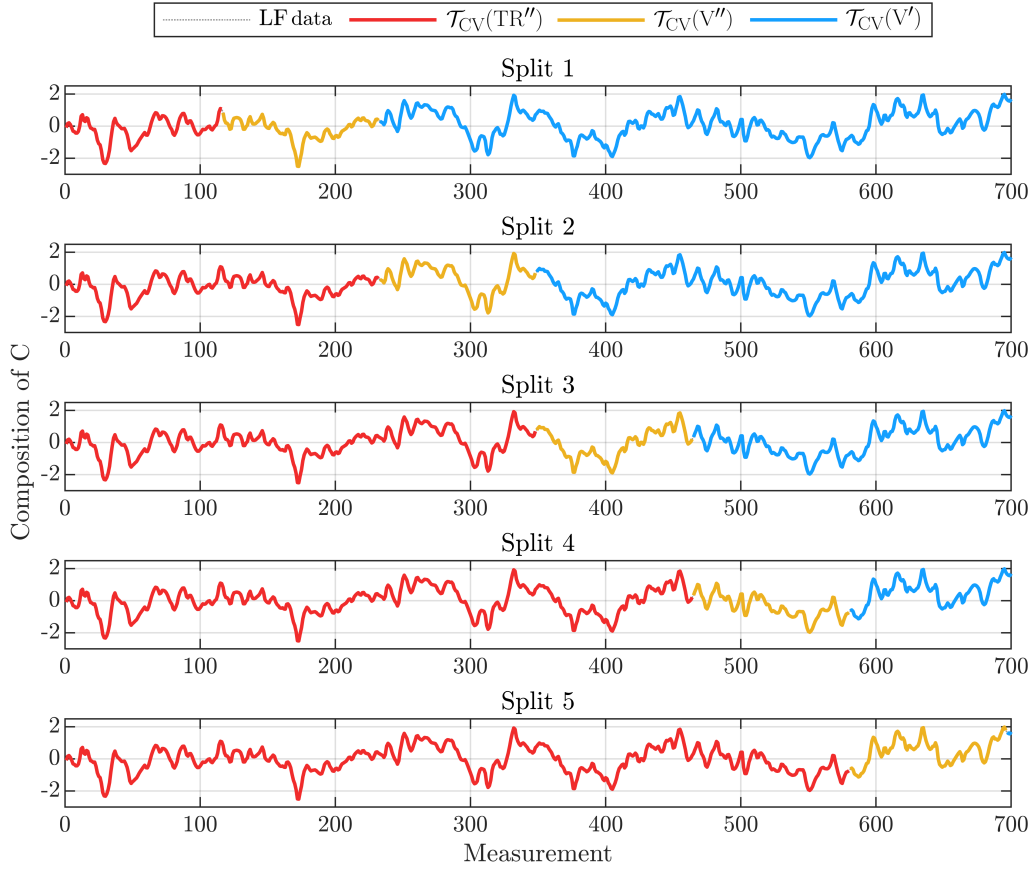


Figure 4: Illustration of five data splits in the expanding-window CV.

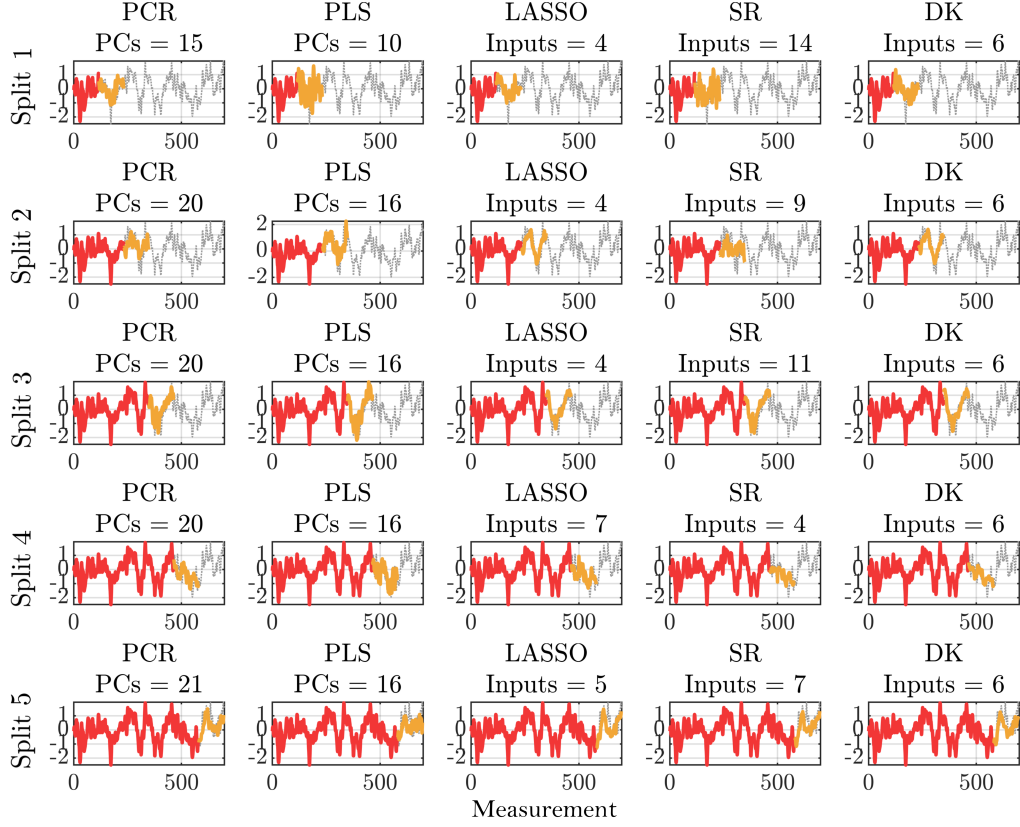


Figure 5: Predicted outputs on validation sets across CV splits for different FS methods. Each plot shows the corresponding number of selected inputs or PCs.

Feature Selection and Cross-Validation

Figure 4 represents five instances of the expanding-window CV data splits on the LF data for the desired output variable. We can observe that the data in all splits vary significantly, which suggests information rich data and gives a good base for fitting a representative model, although, especially for Split 1, only a narrow frequency spectrum seems to be present.

Figure 5 presents the CV procedure in detail showing the results for all splits row-wise and for all methods column-wise. For each training and

Table 1: Average validation RMSE across all splits for used FS methods against LF analyzer data.

FS method	Average $\text{RMSE}_{\text{LF}}^{\mathbf{Y}''}(\hat{y}_{\text{LF}})$
PCR	0.5865
PLS	0.7305
LASSO	0.5954
SR	0.7097
DK	0.5731

validation (red and orange model predictions $\hat{y}_{\text{LF}}^{\text{stat}}$), the number of PCs and inputs is shown. The dimensionality reduction methods (PCR, PLS) use high number of PCs (10–20) compared to other purely statistical methods (LASSO, SR) with 4–14 single-variable inputs.

The DK approach uses six inputs uniformly. Qualitatively, the inputs cover main sections along the material-flow path (Fig. 3). Specifically, the selected variables include: (i) feed composition and (ii) total fresh feed flow rate at DK1 tag, which represent overall throughput; (iii) reactor pressure and (iv) reactor liquid level at DK2 tag that signify the rate of material conversion in reactions. Since direct measurements of product flow are not available in the dataset, (v) recycle flow at DK3 and (vi) total fresh C feed at DK4 are used instead, as they directly monitor material accumulation and flow interactions.

One can see that all the methods are able to capture the data trends although there exist clear qualitative differences not only in the data splits with smaller amounts of training data. This can be seen quantitatively through

Table 2: RMSE comparison of LF data and models against HF data.

Model ξ	$\text{RMSE}_{\text{HF}}^{\text{TR}''}(\xi)$	$\text{RMSE}_{\text{HF}}^{\text{V}''}(\xi)$
y_{LF}	0.9173	0.5462
$\hat{y}_{\text{LF}}^{\text{stat}}$	0.6123	0.4611
$\hat{y}_{\text{LF}}^{\text{dyn}}$	0.6197	0.3879
$\hat{y}_{\text{LF,GP}}^{\text{stat}}$	0.6343	0.5129
$\hat{y}_{\text{LF,AE}}^{\text{stat}}$	0.5167	0.4453
$\hat{y}_{\text{LF,NN}}^{\text{stat}}$	0.7724	0.4290

the average validation $\text{RMSE}_{\text{LF}}^{\text{V}''}$ summarized in Tab. 1, which is a representative aggregate. PCR and LASSO methods perform well, while PLS and SR exhibit higher validation errors. Finally, the DK approach yields the lowest validation error, when compared to purely statistical methods. This suggests consistent input selection. The selected DK inputs are used for single-fidelity model selection.

Single-Fidelity Modeling

We train all variants of presented static models as well as linear dynamic model. For the dynamic model order selection, BIC suggests one input/output lag, while AIC and AICc prefer higher orders. We adopt the BIC choice to keep the model simple.

We compute RMSE against the HF data for all considered models (see Tab. 2) as we ultimately aim to obtain a representation of the HF trend within the later MF model. The table also compares the RMSE to LF data to assess the output prediction by the online analyzer. Results show that all

models, despite being trained on LF data, correlate well with the HF data and would readily improve output monitoring. The models basically smooth out the LF data signal, which then aligns better with the HF data.

Linear dynamic model appears to be a good trade-off between prediction accuracy and complexity that would be added by a nonlinear model. This model is selected as the reference formulation for subsequent MF modeling. The best nonlinear model is achieved using AE, which is noteworthy given that this model does not use the DK features but constructs a well behaved nonlinear latent space. On contrary, NN model shows the largest imbalance among the trained models between the predictions on two datasets.

Kernel Tuning

We train GP models for MF model (29) with different covariance kernels, defined in Section 3.4, to study their effect on prediction consistency. The vector of regressors comprises $\phi_{\text{MF},i} = [\hat{y}_{\text{LF},i}^{\text{dyn}}, \phi_{\text{HF}}^{\text{stat, DK, T}}]^T$. Here, we use the training $\mathcal{T}_{\text{CV}}(\text{TR}')$ and validation $\mathcal{T}_{\text{CV}}(\text{V}')$ subsets. The distribution of validation RMSE values from 100 training runs is given in Fig. 6. Simpler kernels show the lowest median RMSE and the narrowest interquartile range than structured composite kernels. The best-median k_2 (*Scaled RBF*) is selected for the use within the MF approach.

Multi-Fidelity Modeling

We train the aforementioned MF models on data from $\mathcal{T}_{\text{MF}}(\text{TR})$ at HF timestamps using GPR. Figure 7 compares all trained models on training and testing subsets. We can notice overall large variations in the LF data compared to the HF measurements, which causes unreliability of online output

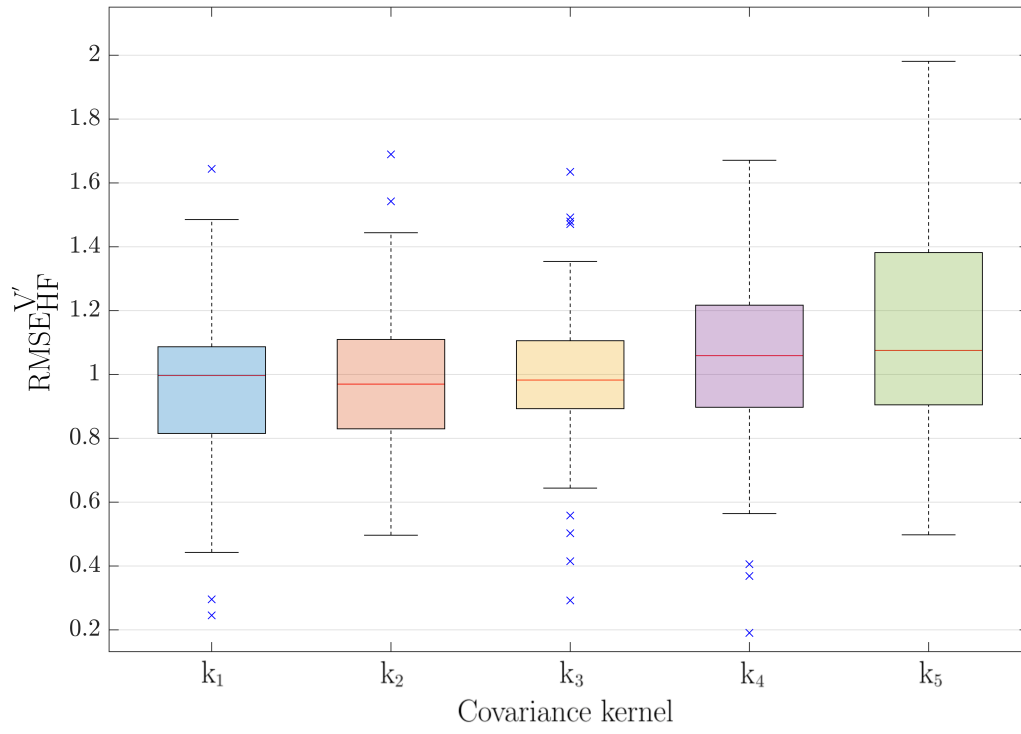


Figure 6: Boxplots of validation RMSE values over 100 training runs of GP model with different kernels.

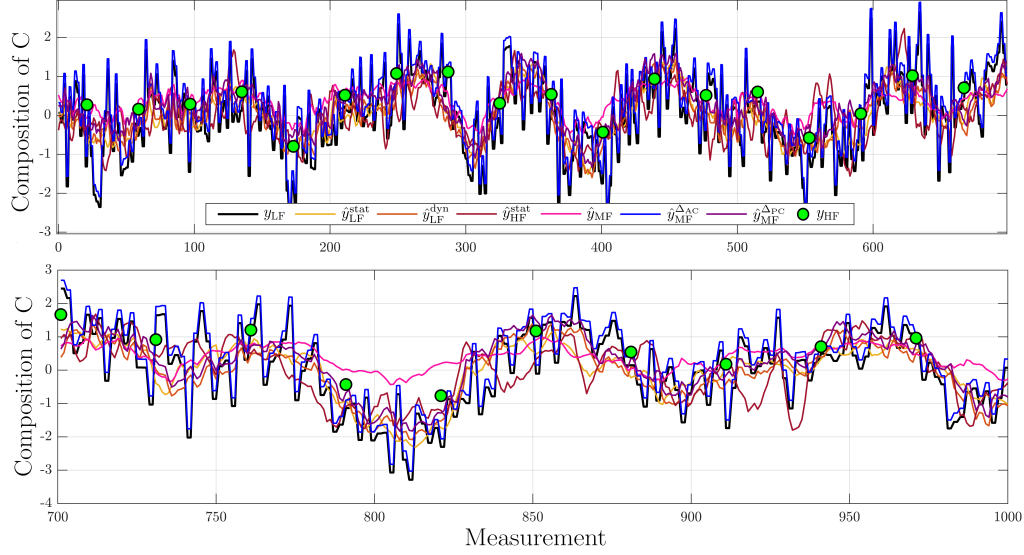


Figure 7: Time-series comparison of data and prediction models on the training (top) and testing (bottom) subsets.

sensing.

We summarize the prediction errors of the trained models in Table 3, where we also assess the percentage of output prediction performance improvement w.r.t. the LF data. Both best-performing LF models provide moderate improvements (about 30 %). This is already a dramatic gain given that these models do not come across the HF data, which is likely to come from the smoothing effect of the models (clearly noticeable in Fig. 7), where we recall that the HF data are generated by filtering the LF signal. The $\hat{y}_{\text{HF}}^{\text{stat}}$ model gives almost 50 % improvement despite showing occasional yet consistent deviations from the main trends (at timestamps 35, 130, 250, 550, 650, 870, 915, 930). Being a linear static model (note that only this type of model can be effectively trained on sparse HF data), the exhibited performance is

Table 3: RMSE performance of all model variants against the HF data. Relative change with respect to the LF analyzer RMSE is shown in parentheses (negative values indicate lower RMSE, i.e., improvement).

Model (ξ)	RMSE_{HF}^{TR} (ξ)	RMSE_{HF}^{TS} (ξ)
y_{LF}	0.8354	0.9770
$\hat{y}_{\text{LF}}^{\text{stat}}$	0.5580 (-33.20 %)	0.6784 (-30.56 %)
$\hat{y}_{\text{LF}}^{\text{dyn}}$	0.5673 (-32.09 %)	0.6810 (-30.30 %)
$\hat{y}_{\text{HF}}^{\text{stat}}$	0.4495 (-46.20 %)	0.5111 (-47.69 %)
\hat{y}_{MF}	0.2463 (-70.52 %)	0.6189 (-36.66 %)
$\hat{y}_{\text{MF}}^{\Delta\text{AC}}$	0.6402 (-23.37 %)	0.8891 (-9.00 %)
$\hat{y}_{\text{MF}}^{\Delta\text{PC}}$	0.4090 (-51.05 %)	0.4827 (-50.60 %)

very good.

While all the single-fidelity models show little overfitting, this aspect significantly varies among the MF models. The MF model \hat{y}_{MF} reaches the lowest training RMSE = 0.2463 among all the trained models, yet the testing RMSE of 0.6189 suggests possible overfitting. The deteriorated performance is noticeable over indices 780–820, where the model fails to respond to the actual variations in the data. The MF model that corrects the analyzer ($\hat{y}_{\text{MF}}^{\Delta\text{AC}}$) appears to be underfitted as it does not even reach the training performance of the HF model. This is clearly visible in the plots, where only a little deviation from LF data is realized by the MF model. This is caused by the LF data trend fluctuations that are hard to correct within the designed feature space. Finally, the $\hat{y}_{\text{MF}}^{\Delta\text{PC}}$ gives the lowest testing RMSE, slightly above 50 % improvement. This is also visible from the prediction trend that follows the

general path of the HF data instead of following the LF fluctuations. Finally, we can say that the best-performing MF model provides only subtle benefits over a much simpler \hat{y}_{HF} model. The reason for this might lie in a reasonable quantity of HF data and little information supplement of LF data over HF data.

The general MF structure defined in Eq. (11) also includes classical autoregressive co-kriging, and related state-of-the-art MF formulations. For completeness, we evaluated a co-kriging formulation as a baseline within this framework. We observed negligible benefits of this strategy, while the performance matched HF model performance. We attribute this phenomenon to the relatively small number of HF samples.

6. Case Study 2: Alkylation Unit in Petrol Refinery

We illustrate the methodology using a Stratco alkylation unit at Slovnaft, a.s. refinery. The unit produces high-octane branched isoparaffins, known as alkylate.

6.1. Process Overview

The Stratco process is divided into three sections: mixing, reaction, and separation (see Fig. 8). In the mixing section, light olefins (mainly propylene and butylene — $\text{C}_3\text{--C}_4$) are blended with two i-C_4 recycle streams, one from downstream distillation and the other from the main separator. These three streams form the reactor feed. In the reaction section, the feed is typically catalyzed by hydrofluoric or sulfuric acid inside the contactor. The olefin is first protonated by the catalyst, forming a reactive carbocation. The carbocation reacts with excess isobutane to stabilize its structure and combines

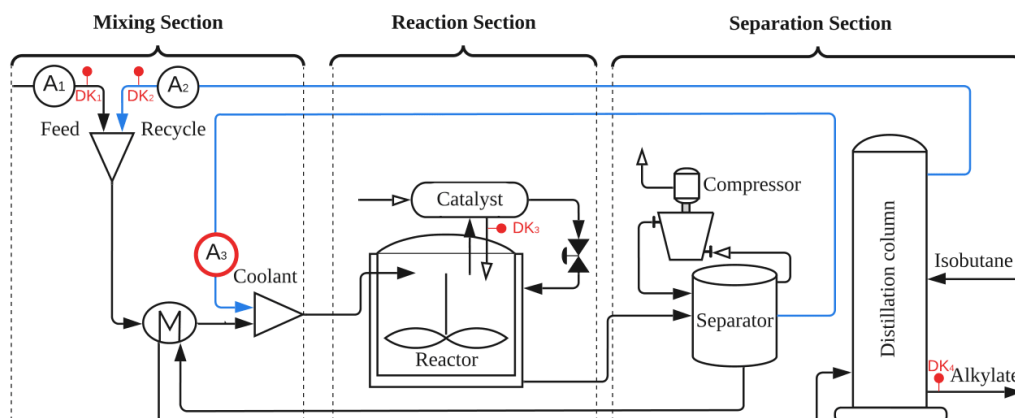


Figure 8: Simplified schematic of the Stratco alkylation unit, highlighting the main process sections (mixing, reaction, and separation) with the placement of key analyzers (A_i) and selected relevant process variables (DK_i).

with another olefin molecule to produce the alkylate — primarily 2,2,4- and 2,3,3-trimethylpentane.

A high isobutane-to-olefin ratio suppresses polymerization into low-octane by-products. The industrial design uses three parallel reactors, each with an acid settler for phase separation and catalyst recovery. Because alkylation is strongly exothermic, the reactors are coupled with a refrigerant loop that maintains a temperature of 5–10 °C. In the separation section, hydrocarbons leaving the settlers pass through a separator, where gases are removed and liquids are sent to fractionation. The distillation column recovers $i\text{-C}_4$ for recycling and delivers alkylate as the final product. Fresh $i\text{-C}_4$ is also fed into the distillation system to maintain the process balance.

Unreliable $i\text{-C}_4$ measurements (by the analyzer A_3) lead to excessive recycling of isobutane, which increases the load on downstream separation units and lowers overall efficiency. The lack of trustworthy online information also

delays the implementation of Advanced Process Control (APC). Operators depend on frequent laboratory analyses, which increases the overall operational costs. These issues motivate the development of a more accurate estimate of the i-C₄ concentration in the recycle stream to maintain the required isobutane-to-olefin ratio. The objective of this study is to develop a soft sensor that reduces the reliance on manual sensor calibration and frequent laboratory validation, while lowering operator workload and process expenses.

6.2. Dataset and Signals

The alkylation unit is equipped with a dense network of sensors measuring flows, temperatures, and pressures, along with several online analyzers that provide composition data. The industrial dataset consists of $n_{\text{LF}} = 25,000$ online measurements and $n_{\text{HF}} = 28$ laboratory analyses, which represents a larger portion of LF data compared to the 1st case study. The available online analyzer y_{LF} (A₃ in Fig. 8) frequently deviates from laboratory values y_{HF} . Two types of discrepancies are observed: sudden outliers and slow drifts. These effects reduce the reliability of the analyzer for process control. The red markers (DK₁–DK₄) represent the location of key process variables.

We partition \mathcal{T}_{LF} into subsets following the systematic chronological split described in Sec. 3.2. The $\mathcal{T}_{\text{MF}}(\text{TR})$ subset contains 13,250 LF samples and 18 HF samples, corresponding to approximately 60 % of the data, while the $\mathcal{T}_{\text{MF}}(\text{TS})$ subset contains 9,250 LF samples and 10 HF samples.

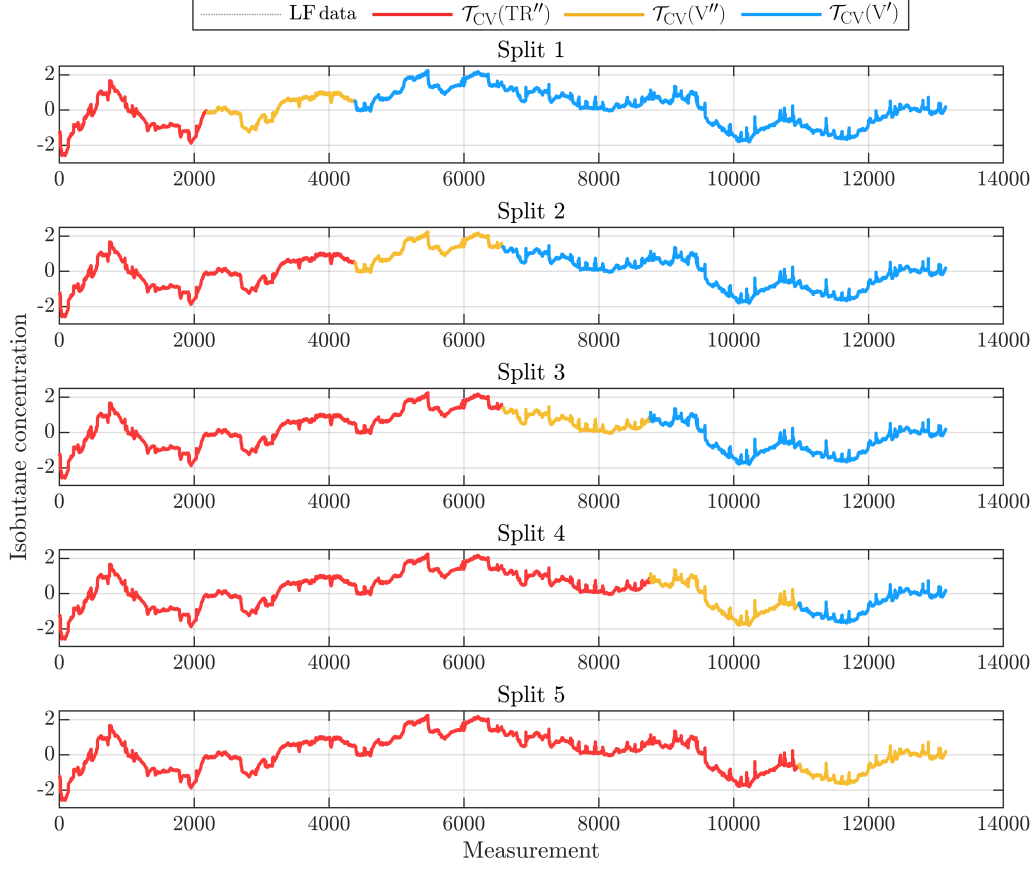


Figure 9: Illustration of the expanding-window cross-validation on five data splits.

6.3. Results

The results are obtained on a preprocessed dataset obtained using the MCD approach described in Sec. 3. From the initial 25,000 LF measurements, 2,500 samples were identified as outliers and removed, resulting in the final dataset of $n_{\text{LF}} = 22,500$ samples used for model training and evaluation.

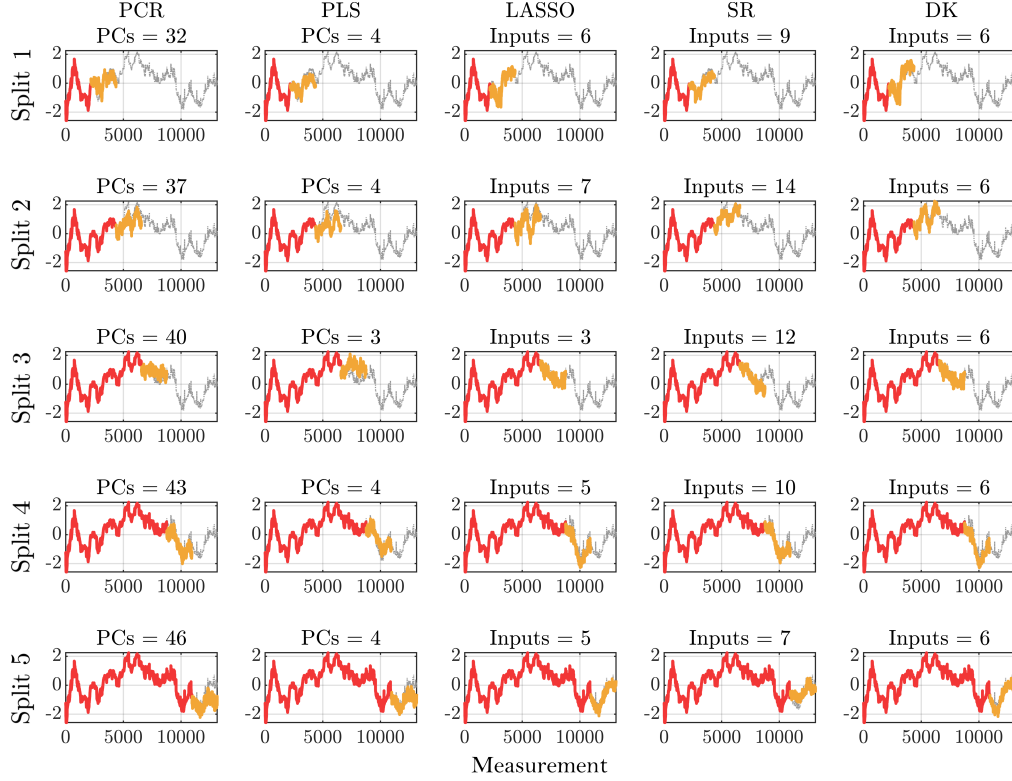


Figure 10: Predicted outputs on validation sets across CV splits for different FS methods. Each plot shows the corresponding number of selected inputs or PCs.

Feature Selection and Cross-Validation

For single-fidelity FS, using an expanding-window CV for LF and k -fold CV for HF data, successive training windows are enlarged four times by approximately 2,000 samples while the validation segments are shifted forward in time (see Fig. 9). We can observe varying plant excitation along the different validation data sections and lower data quality evidence by several observable short-term spikes attributed to the analyzer trips.

The CV procedure is visualized in Fig. 10 and the average validation RMSE values are summarized in Tab. 4. Complexity of the trained mod-

Table 4: Average validation RMSE across all splits for used FS methods against LF analyzer data.

FS method	$\text{RMSE}_{\text{LF}}^{\mathbf{Y}''}(\hat{y}_{LF})$
PCR	0.5180
PLS	0.6050
LASSO	0.3314
Stepwise	0.3543
DK	0.2851

els varies significantly (compared to the 1st case study) both among the FS methods and among the different data splits, owing to the aforementioned varying excitation levels of the LF signal and the overall signal quality. PLS and LASSO methods yield the simplest models, 3–4 PCs and 3–7 inputs, respectively, although only the LASSO method gives a good prediction model, i.e., the best pure-statistics-driven prediction model judged by the average prediction errors. Predictions by PCR require a large number of PCs (32–46), which points to the complexity of the dataset. The prediction quality is moderate yet this performance is unsatisfactory given the model input size. The SR method achieves average prediction accuracy and model complexity. The most consistent results are obtained when we use the DK features. The selected inputs include four concentration variables from the feed (DK₁) and recycle (DK₂) streams in the mixing section, together with the recycled catalyst flow rate (DK₃) and the alkylate product flow rate (DK₄).

Table 5: RMSE comparison of LF data and models against HF data.

Model (ξ)	RMSE_{HF}^{TR'} (ξ)	RMSE_{HF}^{V'} (ξ)
y_{LF}	0.5071	0.6898
$\hat{y}_{\text{LF}}^{\text{stat}}$	0.2207	0.6275
$\hat{y}_{\text{LF}}^{\text{dyn}}$	0.4467	0.6152
$\hat{y}_{\text{HF,GP}}^{\text{stat}}$	0.5296	0.7007
$\hat{y}_{\text{HF,AE}}^{\text{stat}}$	0.5337	0.7045
$\hat{y}_{\text{HF,NN}}^{\text{stat}}$	0.5144	0.6482

Single-Fidelity Modeling

We compare all variants of presented static models and linear dynamic model using the Split 3 of the CV, which is representative of the overall dataset. For the selection of dynamic model order, BIC indicates a second-order model, while AIC and AICc suggest significantly higher order of 14. To avoid overfitting, we adopt the BIC-based guideline which is capable of representing the transport delay between the reaction and separation sections and the analyzer A₃ on the recycle line.

Trained models performance is evaluated against the HF data to assess how well LF-driven models learn the underlying process trend. Table 5 summarizes the RMSE_{HF}^{V''} values. Linear LF models can already slightly reduce the error relative to the analyzer signal. Although the error over the first data partition is higher for the dynamic model, its predictions are smoother than the predictions of the static model and less sensitive to short-term analyzer fluctuations. The trained nonlinear static models exhibit increased sensitiv-

ity to LF data fluctuations and try to explain nonlinear effects in the data trend, leading to relatively high errors over HF data. Although nonlinear models are capable of capturing complex relationships, they do not provide any benefit over linear methods in this case study. Based on the results, we select the dynamic LF model as the most suitable predictor, since it captures the HF trend most consistently and provides smoother behavior than other trained variants. This behaviour indicates that explicit representation of process memory and transport delay is more important than nonlinear static flexibility when the HF signal is limited and the LF analyzer exhibits short-term fluctuations. This is particularly relevant in continuous refinery operation, where residence time and recycle streams introduce delayed responses between manipulated and measured variables.

Kernel Tuning

We apply the kernel tuning procedure to the Alkylation dataset. The vector of regressors for GP models comprises $\phi_{\text{MF},i} = [\hat{y}_{\text{LF},i}^{\text{dyn}}, \phi_{\text{HF}}^{\text{stat, DK}, \top}]^\top$. This combination uses the best-performing LF predictor as the first regressor combined with the static input regressors selected by the DK approach. Figure 11 shows the RMSE distribution across 100 runs for the five candidate kernels. The *Scaled RBF* kernel (k_2) achieves the lowest median RMSE and the narrowest interquartile range. We therefore select kernel k_2 as the final covariance structure for MF model training.

Multi-Fidelity Modeling

The aforementioned MF models are trained and compared to the single-fidelity models in Tab. 6. Linear models generally follow the overall trend of

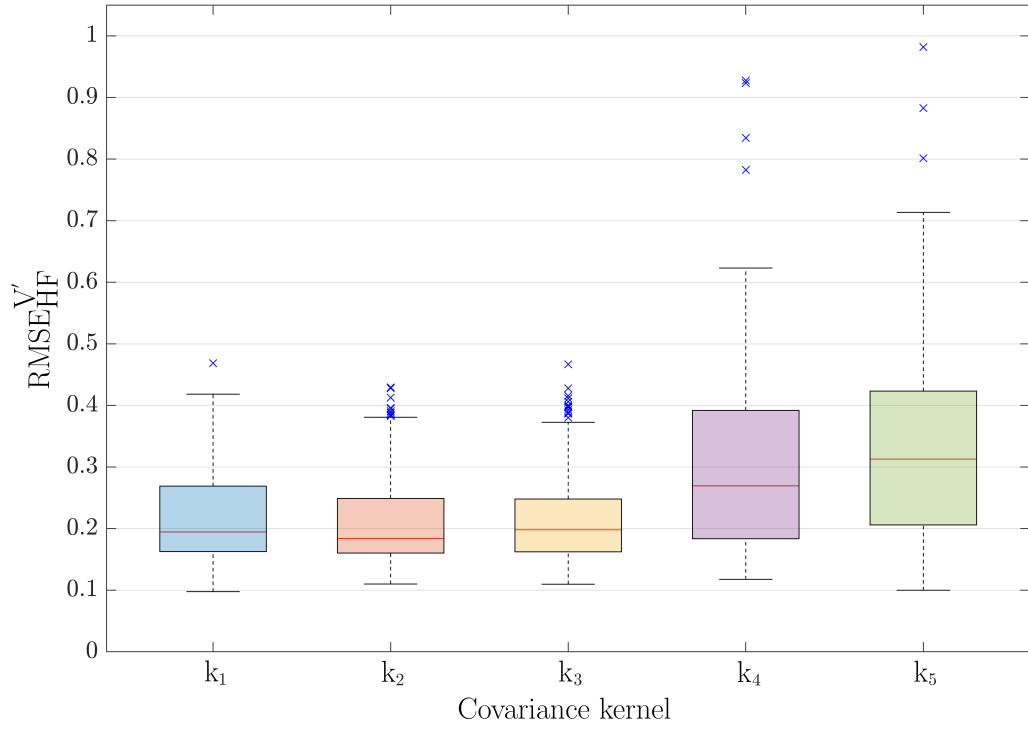


Figure 11: Validation RMSE distributions over 100 iterations of GP model training with different kernels. The boxplots summarize the variability in performance across multiple iterations.

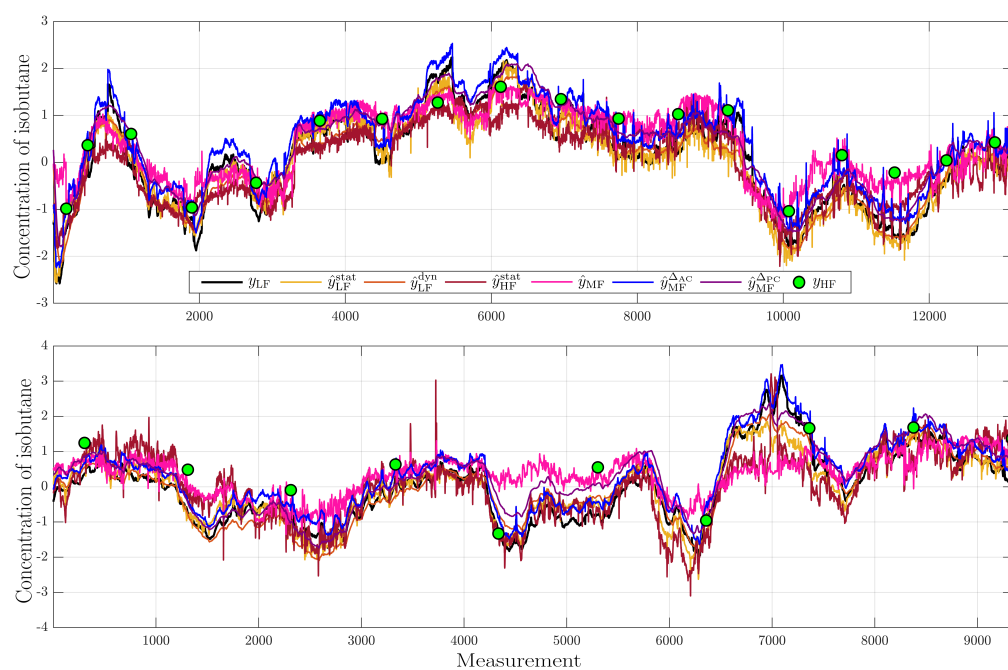


Figure 12: Time-series comparison of data and prediction models on the training (top) and testing (bottom) subsets.

y_{LF} , reproducing signal fluctuation, which is known to be anomalous. Both LF models show worse performance than the analyzer (LF data) itself within testing subset. The HF model $\hat{y}_{\text{HF}}^{\text{stat}}$ exhibits much better prediction accuracy and slightly improves analyzer prediction. However, it shows undesired behavior: over training indices 5000–7000, it deviates from y_{LF} and y_{HF} , while in testing it sometimes closely tracks LF variations (e.g., index 2500) and exhibits spikes or contextual outliers at testing indices 3500 and 7000, likely due to overfitting.

The MF models show very good performance overall and all are capable of improving the online analyzer readings. Similarly to the TEP case study, model \hat{y}_{MF} is prone to overfitting. Its 10-fold increase of RMSE from training to testing evidences this globally, while locally this can be observed as failing to capture local variations around index 4300 on the testing dataset. The remaining two MF models achieve superior performance, providing more than 20% improvement w.r.t. the LF data. Although, the analyzer-based correction MF $\hat{y}_{\text{MF}}^{\Delta\text{AC}}$ increases occasionally the distance from HF data (training indices 5000 and 6000), which might point to local underfitting, it improves local accuracy in testing around the index 4300. The dynamic-prediction correcting MF model $\hat{y}_{\text{MF}}^{\Delta\text{PC}}$ provides the best performance, closely following HF data within 5% deviation and reducing local errors effectively. This model would be clearly beneficial compared to any other alternative.

Overall, the results are consistent with the TEP case study with performance ranking: Analyzer < LF < HF < MF. The $\hat{y}_{\text{MF}}^{\Delta\text{PC}}$ formulation gives the most consistent predictions of the output variable. It appears that analyzing the LF data trends at first and capturing them within a predictive model uncov-

Table 6: RMSE performance of all model variants against the HF data. Relative change with respect to the LF analyzer RMSE is shown in parentheses (negative values indicate lower RMSE, i.e., improvement).

Model (ξ)	RMSE _{HF} ^{TR} (ξ)	RMSE _{HF} ^{TS} (ξ)
y_{LF}	0.6351	0.7945
$\hat{y}_{\text{LF}}^{\text{stat}}$	0.6280 (-1.10%)	0.8534 (+7.41%)
$\hat{y}_{\text{LF}}^{\text{dyn}}$	0.6125 (-3.56%)	0.8222 (+3.48%)
$\hat{y}_{\text{HF}}^{\text{stat}}$	0.4441 (-30.07%)	0.7447 (-6.28%)
\hat{y}_{MF}	0.0722 (-88.63%)	0.7715 (-2.90%)
$\hat{y}_{\text{MF}}^{\Delta_{\text{AC}}}$	0.4301 (-32.27%)	0.6136 (-22.77%)
$\hat{y}_{\text{MF}}^{\Delta_{\text{PC}}}$	0.3698 (-41.77%)	0.6071 (-23.59%)

ers hidden effects of available process readings on the output variable. For comparison, a co-kriging model based on Eq. (11) was also evaluated, however, its performance was not positively outstanding. With only $n_{\text{HF}} = 28$ HF samples, more complex MF models do not provide additional benefit over the simpler model formulations.

7. Discussion and Summary

The results obtained on both the Tennessee Eastman Process (TEP) and the industrial alkylation unit demonstrate that the proposed multi-fidelity (MF) framework provides a consistent improvement over conventional single-fidelity soft sensors. The gains are not uniform across all configurations but follow clear and reproducible patterns. In both case studies, the MF correction reduces systematic bias and long-term drift present in the low-

fidelity (LF) analyzer signals as well as short-term noise. This behavior is reflected in improved prediction performance and in lower overfit ratios. One can align the behavior of MF models as acting like a dynamic bias update of the mis-calibrated online sensors.

Feature selection plays an important role in the observed performance. Regressors selected using domain knowledge (DK) yield consistent and interpretable models across all data splits, with inputs that have clear physical meaning. In contrast, purely statistical selection methods such as LASSO or PLS show higher variability between training and validation subsets. Additional experiments using high-fidelity (HF) feature selection do not result in comparable performance gains. This outcome is primarily caused by the limited number and temporal sparsity of laboratory measurements, which restrict the reliability of HF-driven optimization. In the TEP benchmark, where the HF signal is generated artificially from the same underlying process dynamics, HF-based selection appears slightly more consistent, however, this advantage does not translate into improved generalization and diminishes when evaluated on unseen data. MF modelling paradigm can be even seen as an advanced (high-level) feature selection selecting new feature spaces, similar to PCA, PLS, or AE. The comparison between model structures further clarifies the source of improvement. Static single-fidelity models provide reasonable predictions, however, the dynamic model variation explicitly accounts for transport delays and short-term dependencies, which leads to consistently better validation performance.

Nonlinear single-fidelity models, including Gaussian Process Regression (GPR), autoencoders (AE), and neural networks (NN), do not show system-

atic advantages over simpler linear variants. Although these models often reduce training RMSE, the improvement does not persist on validation or testing subsets. This behavior indicates overfitting and is more pronounced in configurations with limited data. Similar tendencies are observed for simple MF formulations without structural constraints, confirming that increased model flexibility alone is insufficient in this setting.

Within the MF framework, kernel choice affects numerical stability rather than absolute accuracy. Smoother covariance functions lead to better convergence and lower sensitivity to measurement noise, especially when HF samples are sparse. The $\hat{y}_{\text{MF}}^{\Delta\text{PC}}$ configuration consistently provides the best trade-off between accuracy and generalization, reducing RMSE by approximately 20–50 % relative to the currently implemented configuration and achieving the lowest overfit ratios. These improvements are most evident on the testing subset, indicating that the MF correction captures persistent discrepancies between LF and HF measurements.

The MF formulation naturally supports extension to multiple layers of low-fidelity information. Additional LF sources, such as data from first-principles model simulators, can be incorporated as sequential residual corrections.

Given the complexity of GP model used within the MF modelling, updating the MF model, when new HF dataset becomes available, remains a non-trivial task. Although the MF structure itself can be preserved, updating the models requires retraining. Changes in operating conditions or measurement characteristics may invalidate previously selected regressors, making repeated DK feature selection necessary. In practice, retraining can

be scheduled based on performance monitoring, for example when the HF prediction error exceeds a predefined threshold.

These observations indicate that the proposed framework reduces prediction error but does not eliminate the need for expert involvement during model training and updates. The consistent behavior observed across both case studies suggests that the MF concept is transferable, while the effort associated with retraining and feature re-selection must be considered when deploying the method in reliable long-term operation.

To clarify the practical implementation, the final recommended workflow follows a constrained sequential structure: Data preprocessing \rightarrow DK-based feature selection with cross-validation verification \rightarrow Linear dynamic LF predictor modeling (bayesian information criterion-based order selection) \rightarrow GPR-based multi-fidelity predictor correction (predictor correction formulation) with validation-based kernel selection.

8. Conclusion

This work presents a multi-fidelity soft-sensor framework for online correction of industrial sensor measurements. By fusing frequent low-fidelity (LF) analyzer data with sparse high-fidelity (HF) laboratory measurements, the framework achieves higher predictive accuracy than conventional single-fidelity models.

Across two case studies, the multi-fidelity (MF) models reduce prediction error by 20–50% compared to online sensors. Integration of domain knowledge with statistical selection ensures consistent predictions while preventing systematic bias. Nonlinear single-fidelity models offer limited ben-

efits, whereas the GPR-based MF correction effectively captures deviations between LF and HF measurements.

Other state-of-the-art MF models (e.g., co-kriging) were also examined. It was observed that computational demands and sensitivity to sparse HF samples of these models limit their practical deployment in industrial environments — with strongly unbalanced LF/HF datasets.

The framework is generalizable to other processes with similar LF/HF data structures. Future work will focus on incorporating additional LF layers and integrating the MF soft sensors into real-time model predictive control to enhance operational efficiency.

Acknowledgements

This work is funded by the Slovak Research and Development Agency under the project APVV-24-0007, and by the Scientific Grant Agency of the Slovak Republic under the grant 1/0263/25. We acknowledge the financial support provided by the European Union through the NextGenerationEU programme under the Recovery and Resilience Plan for Slovakia, projects no. 09I01-03-V05-00002, no. 09I01-03-V04-00024, and the Early Stage Grant no. 09I03-03-V05 (23-04-06-A).

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The data that has been used is either openly available (1st case study; reference is included) or confidential (2nd case study).

Author contribution

Rastislav Fáber: Conceptualization, Methodology, Software, Investigation, Data Curation, Visualization, Writing - Original Draft, Writing - Review & Editing. Marco Vaccari: Methodology, Validation, Review & Editing. Riccardo Bacci di Capaci: Methodology, Validation, Review & Editing. Karol Ľubušký: Validation, Investigation, Resources, Formal analysis. Gabriele Pannocchia: Methodology, Validation, Review & Editing. Radoslav Paulen: Supervision, Conceptualization, Methodology, Validation, Review & Editing.

References

- Alauddin, M., Khan, F., Imtiaz, S., Ahmed, S., Amyotte, P., 2023. Integrating process dynamics in data-driven models of chemical processing systems. *Process Safety and Environmental Protection* 174, 158–168.
- Armenise, G., Vaccari, M., di Capaci, R.B., Pannocchia, G., 2018. An open-source system identification package for multivariable processes, in: 2018 UKACC 12th International Conference on Control (CONTROL), IEEE. pp. 152–157.
- Averkiev, S., 2018. Tennessee eastman process simulation dataset. Kaggle. URL: [https://www.kaggle.com/datasets/averkij/tennessee-](https://www.kaggle.com/datasets/averkij/tennessee)

- `eastman-process-simulation-dataset`. data referenced in Rieth et al. (2017), "Issues and Advances in Anomaly Detection Evaluation for Joint Human-Automated Systems." Sponsored by the Office of Naval Research.
- Chollet, F., et al., 2015. Keras. URL: <https://github.com/fchollet/keras>.
- Curreri, F., Graziani, S., Xibilia, M.G., 2020. Input selection methods for data-driven soft sensors design: Application to an industrial process. *Information Sciences* 537, 1–17.
- Duvenaud, D.K., 2014. Automatic model construction with Gaussian processes. Ph.D. thesis. University of Cambridge. Chapter 1: Expressing Structure with Kernels.
- Efroymson, M.A., 1960. Multiple regression analysis, in: Ralston, A., Wilf, H.S. (Eds.), *Mathematical Methods for Digital Computers*. Wiley, pp. 191–203.
- Fernández-Godino, M.G., 2023. Review of multi-fidelity models. *Advances in Computational Science and Engineering* 1, 351–400.
- Forrester, A.I.J., Sóbester, A., Keane, A.J., 2007. Multi-fidelity optimization via surrogate modelling. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* 463, 3251–3269.
- Fortuna, L., Graziani, S., Xibilia, M., 2005. Soft sensors for product quality monitoring in debutanizer distillation columns. *Control Engineering Practice* 13, 499–508.

- Fáber, R., Klaučo, M., Ľubušký, K., Paulen, R., 2025a. Integrating linear and nonlinear models for enhanced process monitoring, in: Proceedings of the 2025 25th International Conference on Process Control, IEEE. pp. 1–6.
- Fáber, R., Vaccari, M., di Capaci, R.B., Ľubušký, K., Pannocchia, G., Paulen, R., 2025b. Improving process monitoring via dynamic multi-fidelity modeling. IFAC-PapersOnLine 59, 199–204. 14th IFAC Symposium on Dynamics and Control of Process Systems, including Biosystems DYCOPS 2025.
- Geladi, P., Kowalski, B., 1986. Partial least square regression: A tutorial. Anal. Chim. Acta 35, 1–17.
- Jiang, Y., Yin, S., Dong, J., Kaynak, O., 2021. A review on soft sensors for monitoring, control, and optimization of industrial processes. IEEE Sensors Journal 21, 12868–12881.
- Kadlec, P., Gabrys, B., Strandt, S., 2009. Data-driven soft sensors in the process industry. Computers & Chemical Engineering 33, 795–814.
- Nargesian, F., Samulowitz, H., Khurana, U., Khalil, E.B., Turaga, D., 2017. Learning feature engineering for classification, in: Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI), International Joint Conferences on Artificial Intelligence Organization. pp. 2529–2535.
- Pantelides, C., Renfro, J., 2013. The online use of first-principles models in

- process operations: Review, current status and future needs. *Computers & Chemical Engineering* 51, 136–148. CPC VIII.
- Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E., 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research* 12, 2825–2830.
- Peherstorfer, B., Willcox, K., Gunzburger, M., 2018. Survey of multifidelity methods in uncertainty propagation, inference, and optimization. *SIAM Review* 60, 550–591.
- Raissi, M., Karniadakis, G.E., 2016. Deep multi-fidelity gaussian processes. *arXiv preprint arXiv:1604.07484* .
- Reinartz, C., Kulahci, M., Ravn, O., 2021. An extended Tennessee Eastman simulation dataset for fault-detection and decision support systems. *Computers & Chemical Engineering* 149, 107281.
- Rousseeuw, P.J., Van Driessen, K., 1999. A fast algorithm for the minimum covariance determinant estimator. *Technometrics* 41, 212–223.
- Russell, E.L., Chiang, L.H., Braatz, R.D., 2000. Tennessee Eastman process, in: *Data-Driven Methods for Fault Detection and Diagnosis in Chemical Processes*. Springer, London, pp. 99–108.
- Santosa, F., Symes, W.W., 1986. Linear inversion of band-limited reflection seismograms. *SIAM Journal on Scientific and Statistical Computing* 7, 1307–1330.

- Shah, D., Wang, J., He, Q.P., 2020. Feature engineering in big data analytics for iot-enabled smart manufacturing—comparison between deep learning and statistical learning. *Computers & Chemical Engineering* 141, 106970.
- Volpe, G., Wehr, J., 2016. Effective drifts in dynamical systems with multiplicative noise: a review of recent progress. *Reports on Progress in Physics* 79, 053901.
- Wang, K., Shang, C., Liu, L., Jiang, Y., Huang, D., Yang, F., 2019. Dynamic soft sensor development based on convolutional neural networks. *Industrial & Engineering Chemistry Research* 58, 11521–11531.
- Yin, S., Li, X., Gao, H., Kaynak, O., 2015. Data-based techniques focused on modern industry: An overview. *IEEE Transactions on Industrial Electronics* 62, 657–667.
- Yuan, X., Li, L., Shardt, Y.A.W., Wang, Y., Yang, C., 2021. Deep learning with spatiotemporal attention-based LSTM for industrial soft sensor model development. *IEEE Transactions on Industrial Electronics* 68, 4404–4414.