HAL open science

Reducing the number of experiments required for modeling the hydrocracking process with kriging through Bayesian transfer learning

Loïc Iapteff, Julien Jacques, Matthieu Rolland, Benoît Celse

▶ **To cite this version:**

Loïc Iapteff, Julien Jacques, Matthieu Rolland, Benoît Celse. Reducing the number of experiments required for modeling the hydrocracking process with kriging through Bayesian transfer learning. 2020. hal-02935247v1

HAL Id: hal-02935247

https://hal.science/hal-02935247v1

Preprint submitted on 16 Sep 2020 (v1), last revised 23 Nov 2021 (v2)

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Reducing the number of experiments required for modeling the hydrocracking process with kriging through Bayesian transfer learning

Loïc Iapteff[1,2]
Julien Jacques[2]
Matthieu Rolland[1]
Benoit Celse[1]
[1]IFP Energies nouvelles, Solaize, France
[2]Université de Lyon, Lyon 2, ERIC UR 3083, Lyon, France

Abstract

The objective is to improve the learning of a regression model of the hydrocracking process using a reduced number of observations. When a new catalyst is used for the hydrocracking process, a new model must be fitted. Generating new data is expensive and therefore it is advantageous to limit the amount of new data generation. Our idea is to use a second dataset of measurements made on a process using an old catalyst. This second dataset is large enough to fit performing models for the old catalyst. In this work, we use the knowledge from this old catalyst to learn a model on the new catalyst. This task is a transfer learning task. We show that the results are greatly improved with a Bayesian approach to transfer linear model and kriging model.

Keywords: Transfer Knowledge of Parameters, Regression modeling, Gaussian Process, MCMC algorithm

1 Introduction

1.1 Industrial Challenge

Refineries convert crude oil into usable products, mostly fuels for the transport industry like gasoline, kerosene for planes or diesel, and high purity chemicals that will be used to produce plastics including propylene, butadiene and aromatics. Commercial refining products must adhere to strict quality norms, ensuring proper performance. For example, diesel must have a high enough cetane number and almost no residual sulfur ($< 10ppmS$) in order to prevent acid rain, have a suitable viscosity, and to not freeze in winter. Each product requires a succession of specific treatments. The most important unit operations are distillations to separate the chemical constituents according to their boiling points, and two types of chemical reaction: 1) purification that aims at removing impurities like sulfur, nitrogen, and 2) conversion that changes the chemical structure of the molecules and improve their commercial value. A refinery is thus a complex network of interconnected units. A modern refinery costs in the order of several billion euros, and there is a great financial interest to optimize design and operation. A refinery operation is constantly adapted to maximize revenues depending on the products' market prices, under the constraints of feedstock availability and unit downtime for maintenance or repair.

Reactions take place in presence of a catalyst whose aim is to reduce temperature, pressure and time required to achieve the performance as well as to orientate the reactions toward the desired products (selectivity). For each reaction, several vendors offer their own catalysts. When offering a new catalyst, a vendor must guarantee its performance: a lot of effort of the catalyst development cycle is to be able to predict, as accurately as possible, its performance in the customer's refinery. Performance prediction is based on both experimental data and models. Experiments are very expensive as they must be performed in conditions mimicking those of refineries on a variety of feedstocks that are not commercial and difficult to obtain. The cost for one point is around 10,000 euros. Modeling can be based on either physical approaches or statistical approaches. Physical approaches are possible for simple molecules and when the scientific knowledge is sufficient enough to predict all properties of interest from the chemical composition. In most cases, this is not possible and correlations are built and used. New catalysts are constantly being developed so that each new generation of a catalyst requires a new model that is built from scratch from new experiments. In this paper, we are interested in using previous knowledge on an old catalyst to build a model for a new catalyst with fewer new data points.

This work focuses on the hydrocracking process that converts heavy products from vac-

uum distillation into diesel and lubricants, which are more valuable products. Chemically speaking, it breaks long hydrocarbon chains into smaller bits in the presence of high temperature ($> 380°C$) and hydrogen under high pressure ($\sim$ 150 bars). The catalyst is a solid porous media shaped into rice grain sized pellets that are stacked into quite large reactors ($\sim$ 20 m in heigh by $\sim$ 6 m in diameter). The hydrocracking catalyst is sensitive to impurities like sulfur and nitrogen, and thus a purification pre-treatment in another reactor (with another catalyst) is required. Downstream of the hydrocracking reactor, a distillation column splits the products into several cuts, one being the diesel cut, another one heavier that will be directed to an upgrading unit to produce lubricants. More specifically, we want to predict the density of the diesel cut (DIES_D154 in Table 1) based on information about the feedstock, the operating conditions and some information on the diesel cut.

The challenge in this paper is to build the best predictive model for a new catalyst with the fewest observations on that new catalyst ("new") and having access to many observations on a previous generation catalyst ("old"). For this reason, we will be working on two data sets, one corresponding to the old catalyst, large in size, and the other corresponding to a new generation of catalyst. We want the latter to be as small as possible. This task is known as transfer learning (Pan and Yang 2010).

1.2 Related work on Transfer Learning

Let's define a domain as $\mathcal{D} = (\mathcal{X}, P(\mathcal{X}))$ with $\mathcal{X}$ a feature space and $P(\mathcal{X})$ its probability distribution, and an associated task $\mathcal{T} = (Y, f)$ with f the function used to predict $y \in Y$ given $\boldsymbol{x} \in \mathcal{X}$. We name $\mathcal{D}_s$ and $\mathcal{T}_s$ the domain and task of source data and $\mathcal{D}_t$ and $\mathcal{T}_t$ the domain and task of the target data. There are two distinct areas in Transfer Learning, the Transductive Transfer Learning when $\mathcal{D}_s \neq \mathcal{D}_t$ and the Inductive Transfer Learning when $\mathcal{T}_s \neq \mathcal{T}_t$. For this work, the source data are those from the old catalyst, while the target data are those from the new one. $\mathcal{X}_s = \mathcal{X}_t$ is the space of the 12 features previously presented while $Y_s = Y_t$ is DIES_D154, the variable to be predicted. The values of these 12 features are determined by the refiner, and we can therefore assume that $P(\mathcal{X}_s) = P(\mathcal{X}_t)$. Thus, $\mathcal{D}_s = \mathcal{D}_t$ and $\mathcal{T}_s \neq \mathcal{T}_t$ which means it is an Inductive Transfer Learning problem.
Inductive Transfer Learning is classically split into three categories: transfer knowledge of instances, transfer of features representation and transfer of parameters. The distinction is, as their name suggests, the way in which information is transferred. Literature on transfer learning is abundant and we will only present some of them here. Those interested can read reviews (Pan and Yang 2010 and Tsung et al. 2018), the latter presenting a more statistical point of view.

Table 1: Features description.

Feature	Description
PPH2	Hydrogen (H2) partial pressure. Hydrogen is used to break molecules bonds.
TWABT	Average temperature of hydrocracking reactor (WABT: Weight Average Bed temperature).
FEED_D154	Feedstock relative density measured at $15°C$ with respect to density of water at $4°C$.
FEED_DS05 FEED_DS50 FEED_DS95	Temperature at which a given percentage of the feedstock is evaporated. Simulated distillation (DS) is an indirect measurement of the distillation temperatures based on gas phase chromatography. It produces a curve that we reduce to 3 temperatures at 5, 50 and 95% of evaporation.
FEED_NIT	Nitrogen content in feedstock. Nitrogen is a catalyst inhibitor found in crude oil.
FEED_SULF	Sulfur content in feedstock. Sulfur is a pollutant causing acid rains after its combustion in engines.
DIES_DS05 DIES_DS50 DIES_DS95	Temperature at which 5, 50 and 95% of the diesel output is evaporated (see FEED_DS for details).
X370+	Percentage of conversion of the feedstock 370+ cut. It indicates how much of hydrocracking was performed.
DIES_D154	Relative density of diesel measured at $15°C$ with respect to density of water at $4°C$. The feature to be predicted.

The transfer knowledge of instances is based on the direct reuse of observations from the source dataset. To quote some works of this method, we can mention Transfer Adaptative Boosting (Dai et al. 2007), which adapts the Adaptative Boosting algorithm for Transfer Learning in classification or its adaptation for regression (Pardoe and P. Stone 2010). As with the AdaBoost algorithm, the observations are weighted. Both source and target data are used. For the target data, the evolution of the weights at each iteration is the same as for AdaBoost. Conversely, for the source data, the weight is increased if the observation is well predicted and therefore considered useful for learning, and it is otherwise decreased. At each iteration, the learning is performed on the source and target data but the error is calculated on the target data only. Another work on instance transfer uses a method that generates more target data (Salaken et al. 2019). The algorithm produces new target observations and can be used when $\mathcal{X}_s = \mathcal{X}_t$. Source domain is abstracted using autoencoder to obtain a new domain with the same number of features as the target domain. Then, clustering is performed on the source domain to have as many clusters as the target data. Each cluster center is associated with a target point and the distribution of data around these centers is reapplied around the associated target point.

For transfer knowledge of features representation, the problem is similar to multitask learning, where the aim is to find the common feature representation (Argyriou, Evgeniou, and Pontil 2007; Argyriou, Pontil, et al. 2008), which can be applied for transfer learning. As it looks for a low dimensional representation shared by the different tasks, this method is more useful when the number of features is large. The proposed algorithm by Argyriou, Evgeniou, and Pontil 2007; Argyriou, Pontil, et al. 2008 works in two alternating steps. The first independently learn parameters of different tasks models. The second aims at keeping the tasks coupled by learning common features across the tasks in an unsupervised way.

The third method is the transfer knowledge of parameters. For Random Forest models, two algorithms can be cited (Segev et al. 2016): Structure Expansion Reduction (SER) and Structure Transfer (STRUT). They both transform the forest learned on source data, tree by tree. SER is a two step algorithm: expansion and reduction. Expansion extends the tree by constructing new sub-trees starting from leaves with target data that reached them. Reduction runs through all internal nodes and transforms them into a leaf if the error is reduced compared to the empirical error of its sub-tree. The STRUT algorithm works on the thresholds by readjusting them on the target data, keeping the feature on which the separation is made. For linear models, one possibility is to find a link between the source and the target model by assuming that some parameters are unchanged or subject to the same transformation (Bouveyron and Jacques 2010). This makes it possible to reduce the

number of parameters to be estimated and maintains knowledge of the source data. Another approach has been proposed to transfer knowledege of parameters in Support Vector Machine models (Evgeniou and Pontil 2004). The idea is to assume that the weighted vector of source and target tasks can be decomposed into two terms, one common to both, the other different.
Independently from the type of model, Bayesian approaches are often considered for transfer knowledge of parameters. For instance, Raina, Ng, and Koller 2006 constructs a Gaussian informative prior to the document classification task using logistic regression. They construct a non-diagonal covariance matrix for the Gaussian prior using auxiliary document classification tasks. The objective is to find the correlated words and thus the correlated parameters instead of assuming their independence. Still with a Bayesian approach, Launay, Philippe, and Lamarche 2015 use a hierarchical Bayesian model to forecast electricity load using a small dataset. The prior distributions are constructed using another dataset, which is larger, on which the model is known to be effective. The parametric regression model is non-linear and a Gaussian prior is chosen. Another paper proposes to solve regression problems in an adaptative way via the Gaussian Process (B. Cao et al. 2010). It assumes that the two Gaussian Process regression models for the source and target tasks share the same parameters θ in their kernel functions. The dissimilarity between source and target task is represented by a parameter that follows a Gamma distribution and thus the model automatically learns the similarity between the tasks.

As our features are already selected, the transfer knowledge of feature representation method is not suitable. We opted for transfer knowledge of model parameters as we consider it to be well suited to the transformation of models that have proven efficient on the source data.

1.3 Our datasets

The datasets are composed of data originating from plants in the form of time series, with measurements being performed daily. For the "old" catalyst set, the data came from 8 plants for a total of 3,177 observations. For the "new" catalyst, the data came from 2 plants for a total of 1,004 observations.
Experts selected 12 features presented in Table 1 that are defined bellow, starting from the top. The first two features are process parameters: partial pressure of hydrogen (PPH2) and temperature (WABT). Next are four properties of the feedstock, namely density (FEED_D154) and temperatures for which 5, 50 and 95% of the feedstocks are evaporated (FEED_DS05, 50 and 95). Each refinery has its own definition for the diesel

distillation cut, some cut higher and some lower, which simply means that the distillation temperatures are different. This, of course, has an impact of the diesel density (a higher cut diesel has more molecules that vaporizes at high temperature and has thus a higher density). Therefore, information on the diesel distillation is included in the model through the DS features (DIES_DS05, 50 and 95). The last feature is the conversion (X370+) which is the fraction of the heavy products that have been hydrocracked. Conversion is, of course, a consequence of the process parameters and feedstock properties. Its prediction is modeling work in itself (see for example Becker, Celse, et al. 2016). All of those features are quantitative variables, including the output to be predicted, and we want to build a regression model:

$$y_i = f(\boldsymbol{x}_i) + \epsilon_i \tag{1}$$

where y_i is the diesel density (DIES_D154) and $\boldsymbol{x}_i = (x_{i,1}, \ldots, x_{i,12})$ is the vector of the 12 features presented above for observation i. Observations are here defined as data points from plant operation track logs and are a series of successive observations labeled by date. The temporal aspect is not considered and each point is assumed to be independent of the others.

The two datasets used in this study are different enough that the feature correlation matrix are different for the two catalysts (Figure 1). Correlations between input features and the property to be predicted (DIES_D154) are slightly distinct. For the old catalyst

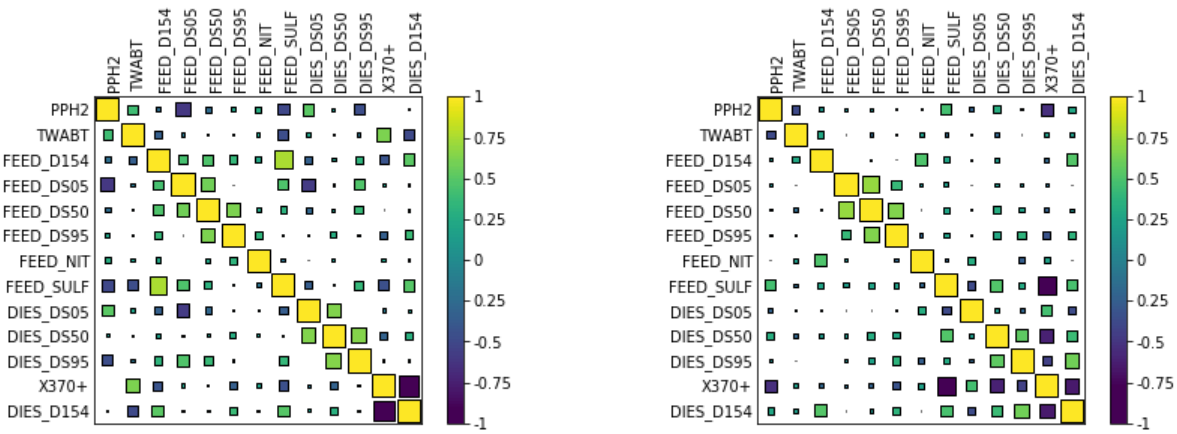

Figure 1: Correlation matrix for two different catalysts. Left is the "old" catalyst (3,177 observations) and right is the "new" catalyst (1,004 observations).

(left), the largest correlation is with conversion (X370+), while the others are at least

twice lower. For the new catalyst (right), the correlations are more homogeneous and new correlations appear as with DIES_DSxx. It is interesting to note that the correlation between input features can be high. For example, feedstock density (FEED_D154) is related to the distillation parameters (FEED_DSxx). Feed sulfur content and feed density are both a consequence of the origin of the crude oil. Similarly, a higher feed density will probably result in a higher cracked diesel density. It is important to note at this point that although correlated, those parameters do not carry the same information. It is also worth noticing that correlation between features depends on the dataset. For example, FEED_D154 is correlated with FEED_SULF for the first catalyst but not the second. The same can be said for FEED_DS05 and PPH2. This can be explained by both the unrepresentative data sampling on a small number of plants, and by a different performance of the catalyst that requires different tuning of the hydrocracking unit, and thus mathematically changes the relationship between feed, products properties and process parameters.

1.4 Content of the paper

In the next section, we start by building a good predictive model for the source dataset (Section 2). Different models are tested and two are chosen to be transferred: linear model for its simplicity and kriging for its good predictive results. For transferring the linear regression model, two methods are used. The first one is a parametric approach (Section 3.1.1) directly inspired from Bouveyron and Jacques 2010. The second approach is Bayesian (Section 3.1.2) and consists in using the source parameter as prior information for the target model. For the kriging model, a Bayesian transfer learning approach is proposed in Section 3.2, which is the main methodological contribution of the paper. This last method offers excellent results and allows the number of target observations required to be greatly reduced. Section 4 concludes the paper with some perspectives.

2 Regression models for the source data

The goal of this paper is to propose transfer learning methods for transferring regression models from an old catalyst to a new one. Obviously, the transfer method depends on the type of regression model. Consequently, several regression models have been tested on the source data: Linear Model (LM), Random Forest (RF), Gradient Boosting (GB), Multilayer Perceptron (MLP), Support Vector Regression (SVR) and Kriging (Cressie 1990). Only two of them will be considered for transfer learning. While most of these models are

well known, Kriging may be the least known and we will present it before detailing the results and retained models.

2.1 Kriging

The Kriging model can be written as follows:

$$y_i = m(\boldsymbol{x}_i) + \delta(\boldsymbol{x}_i) \tag{2}$$

where $m(\boldsymbol{x}_i) = \Sigma_{j=0}^{d} \beta_j f_j(\boldsymbol{x}_i)$ is the trend of the model and $\delta(\boldsymbol{x}_i)$ is a second-order Gaussian stationary process such that:

$$cov(\delta(\boldsymbol{x}_i), \delta(\boldsymbol{x}_{i'})) = C(|\boldsymbol{x}_i - \boldsymbol{x}_{i'}|).$$

Its covariance depends only on the distance between the observations $\boldsymbol{x}_i$ and $\boldsymbol{x}_{i'}$.

The shape of the trend is assumed to be known, i.e. the f_j are known, and only β_j should be estimated. In this work, f_j are chosen to be the identity function for $j > 0$, $f_0(\boldsymbol{x}_i) = 1$ and $d = p$ the number of features, which is a common choice corresponding to a linear trend. The shape of the covariance function is also assumed to be known:

$$\begin{aligned} C(h_1, \ldots, h_p) &= \sigma^2 R_{\boldsymbol{\theta},\tau}(h_1, \ldots, h_p) = \sigma^2 \prod_{j=1}^{p} g(h_j, \theta_j) + \delta_{\mathbf{0}}(h_1, ..., h_p)\tau, \\ \delta_{\mathbf{0}}(h_1, ..., h_p) &= \begin{cases} 1 \text{ if } (h_1, ..., h_p) = (0, ..., 0), \\ 0 \text{ else.} \end{cases} \end{aligned}$$

with σ^2 the variance of the process, τ is a constant introduced to treat discontinuity known as the nugget effect (Cressie 1988), and $\boldsymbol{\theta} = (\theta_j)_{j=1,\ldots,p}$ the parameters of the covariance function g, which is chosen to be the Matern 5/2 covariance function: $g(h_j, \theta_j) = (1 + \sqrt{5}|h_j|/\theta_j + 5h_j^2/(3\theta_j^2)) \exp(-\sqrt{5}|h_j|/\theta_j)$ (Stein 2012). These are classic choices (Roustant, Ginsbourger, and Deville 2012).

Parameter estimation is traditionally done by maximum likelihood, and prediction for a new observation $\boldsymbol{x}_0$ with the kriging model is achieved by looking for the best linear unbiased predictor $\widehat{y}_0$. It is obtained by finding $\boldsymbol{\lambda}(\boldsymbol{x}_0) = (\lambda_1(\boldsymbol{x}_0), \ldots, \lambda_n(\boldsymbol{x}_0))^T$ minimizing $\mathbf{E}[(\delta(\boldsymbol{x}_0) - \Sigma_{i=1}^{n} \lambda_i(\boldsymbol{x}_0)\delta(\boldsymbol{x}_i))^2]$. The solution is given by $\widehat{\boldsymbol{\lambda}}(\boldsymbol{x}_0) = \boldsymbol{C}^{-1} c(\boldsymbol{x}_0)$ with $\boldsymbol{C} = (C(|\boldsymbol{x}_i - \boldsymbol{x}_{i'}|))_{1 \leq i, i' \leq n}$ and $c(\boldsymbol{x}_0) = (C(|\boldsymbol{x}_0 - \boldsymbol{x}_i|))_{1 \leq i \leq n}$.
Finally, the prediction is:

$$\begin{aligned} \widehat{\boldsymbol{\beta}} &= (f(\boldsymbol{X})^T \boldsymbol{C}^{-1} f(\boldsymbol{X}))^{-1} f(\boldsymbol{X})^T \boldsymbol{C}^{-1} \boldsymbol{y}, \\ \widehat{y}_0 &= f(\boldsymbol{x_0})\widehat{\boldsymbol{\beta}} + c(\boldsymbol{x_0})^T \boldsymbol{C}^{-1}(\boldsymbol{y} - f(\boldsymbol{X})\widehat{\boldsymbol{\beta}}), \end{aligned}$$

where $\boldsymbol{X}$ is the matrix whose n rows are the $\boldsymbol{x}_i$ and $\boldsymbol{y} = (y_i)_{i=1,\dots,n}$.

2.2 Experimental settings

In order to reduce the impact of outliers, the source data are normalized according to the median for centering and the interquartile range for reduction. To evaluate model quality, the source dataset is split into a training and test set according to the Kennard and Stone algorithm (Kennard and L. Stone 1969). This is an iterative algorithm. It is initialized with a training dataset composed of the two most distant points of the complete dataset and a test set composed of the remaining points. Then, at each iteration one point $\boldsymbol{x}_i$ is moved from the test dataset to the training dataset such as $\boldsymbol{x}_i = \underset{\boldsymbol{x}_i \in \text{Test}}{argmax}(\underset{\boldsymbol{x}_{i'} \in \text{Training}}{min} dist(\boldsymbol{x}_i, \boldsymbol{x}_{i'}))$ until the training dataset reaches a pre-defined size. This ensures a training set with the maximum amount of information. The sizes of the test and train sets are both 50% of the complete dataset size. The training and test sets are no longer normalized, and a second normalization is performed on the training set. Parameters of this normalization are then used to normalize the test set. For the section 3, the target dataset is also normalized according to these parameters. To complete preprocessing, an outlier detection is achieved on training set using the Local Outlier Factor method (Breunig et al. 2000), with 10 nearest neighbors. The threshold is chosen such that the observations with a distant LOF score compared to the other are ranked as outlier. This leads to a threshold of 1.5 and about 2.5% of the observations are classified as outliers.

RF, GB, SVR and MLP have hyperparameters to tune: number of trees and their depths for RF and GB, number and size of the layers and activation function for MLP, size of the epsilon-tube and regularization parameter for SVR. A grid search for these hyperparameters is used with a range specified in Table 2. For each model, the hyperparameter combination that minimizes the RMSE score ($\sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}$ where $\hat{y}_i$ is the predicted value) evaluated with 10 fold cross-validation is selected. For all these models (RF, GB, SVR and MLP), the scikit-learn implementation is used (Pedregosa et al. 2011).

For the kriging model, the DiceKriging package from R (Roustant, Ginsbourger, and Deville 2012) is used to fit the model. This package uses the BFGS algorithm to maximize the likelihood. The chosen hyperparameters are mentioned in section 2.1. As a nugget effect is used, it has to be estimated. Universal kriging is considered, it means that the trend parameters are unknown and are estimated. The upper bound of the $\boldsymbol{\theta}$ parameters is set to twice the difference between the maximum and minimum values for each feature, as suggested in Roustant, Ginsbourger, and Deville 2012.

Table 2: Hyperparameters and their values tested by grid search for the different models.

Model	Hyperparameter	Value
RF	n_{tree}	$\{10k\}_{k=1,\ldots,10}$
	$\max_{depth}$	$\{5k\}_{k=1,\ldots,6}$
GB	n_{tree}	$\{10k\}_{k=1,\ldots,10}$
	$\max_{depth}$	$\{5k\}_{k=1,\ldots,6}$
MLP	n_{layers}	$\{1, 2, 3\}$
	size_{layer}	$\{5, 8, 12, 16\}$
	f_{activ}	$\{relu, tanh\}$
SVR	ϵ	$\{10^{-4}, 10^{-3}, 10^{-2}\}$
	C	$\{1, 5, 10, 15, 20, 25, 30\}$

2.3 Model comparison

The models are evaluated on the test set according to three criteria: RMSE score, percentage of observations for which the prediction error is less than 0.005 and 0.0025. The limit of 0.005 on the prediction error is commonly used in the field of oil product density prediction, knowing that experimental measurement accuracy of DIES_D154 is 10 times smaller (0.0005).

The results are given in Figure 2 and Table 3. Globally, the results are quite similar for all the models. For the three studied scores, the kriging model offers the best results with a RMSE score below 0.0023. The RF, MLP, SVR and GB models offer slightly poorer results and are all similar with RMSE scores between 0.0024 and 0.0026. The percentage of observations with an error lower than 0.005 is higher than 95% for these 5 models, which is quite satisfactory indeed. Although satisfactory, the results of LM are worse than those of the other models and have a bias by plant. However, it has the advantage of being an easily interpretable and understandable model, allowing the effects of each variable to be analyzed. Let finally notice that the prediction qualities per plant are nearly equivalent. This can be seen in Figure 2 where the colored dots represent the plants, named from A to H.

Since the kriging model provides the best performance, it is chosen as a candidate to be transferred to a new catalyst. Additionally, we also select the linear model for its simplicity and interpretability. The next section describes transfer learning methods for the linear and kriging models.

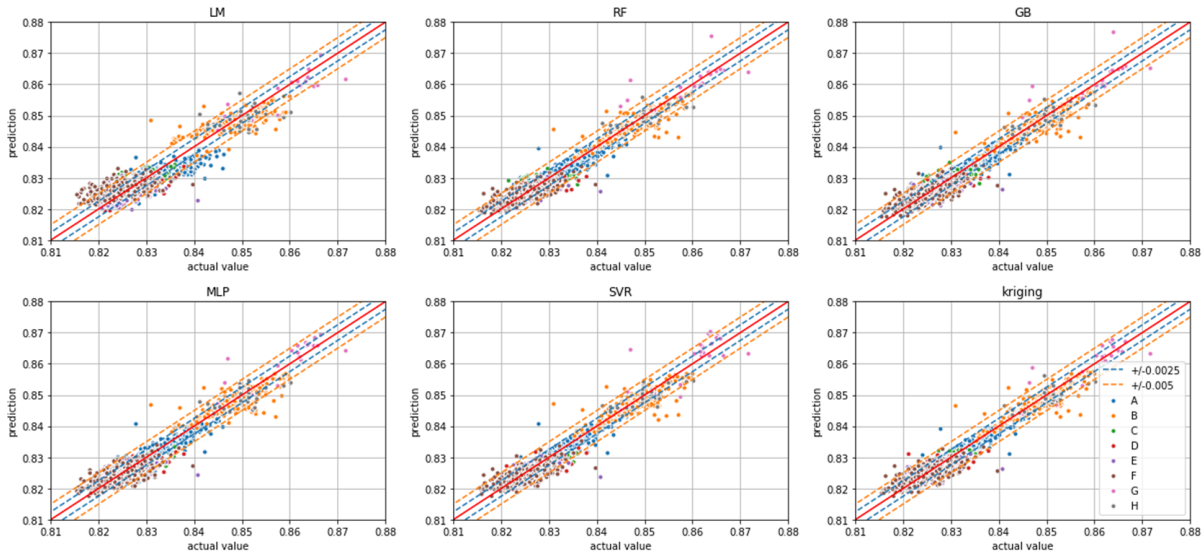

Figure 2: Results of the different models fitted on the training source data, applied to test source data. The colored dots represent different plants.

3 Transfer Learning for regression

In this section, a Bayesian transfer learning method is proposed for the kriging model. For comparison, two transfer learning methods, Bayesian and parametric, are also considered for the linear model. The target dataset on which the models have to be transferred is composed of 1,004 observations. Since, in practice, the goal is to be able to build a model with as few points as possible from the target catalyst, a subset of size n_t is randomly extracted from the whole target dataset. Different sizes of n_t will be considered, and for each size, 10 randomly sampled datasets are considered and average results are presented. The quality of the model is evaluated with the RMSE score evaluated on an independent test set composed of 804 data points (different from the n_t data used for transfer). In the present industrial context, a model is considered satisfactory if the RMSE score is lower than 0.005. For memory, the respective RMSE scores on the source dataset for the linear and kriging models are 0.0033 and 0.0023 (Table 3), respectively, which is quite satisfactory.

Variables referring to the source will be indexed by "s" ($\boldsymbol{X}_s$, $\boldsymbol{y}_s$,...) while those referring to the target will be indexed by "t" ($\boldsymbol{X}_t$, $\boldsymbol{y}_t$,...).

Table 3: Scores of different models fitted on the training source data, applied to the test source data.

Model	LM	RF	GB	MLP	SVR	Kriging
RMSE	0.00331	0.00248	0.00243	0.00259	0.00244	0.00229
+/- 0.0025	59.9%	76.5%	76.8%	73.0%	78.6%	80.5%
+/- 0.005	87.3%	95.1%	95.5%	95.0%	95.5%	96.2%

3.1 Transfer learning for the linear model

The model for the source catalyst is

$$y_i = \beta_{s0} + \sum_{j=1}^{p} \beta_{sj} x_{ij} + \epsilon_i \tag{3}$$

with $\epsilon_i \sim \mathcal{N}(0, {\sigma_s}^2)$ and $p = 12$. The maximum likelihood estimator on the source data give the results in Section 2.3. Our goal is to estimate the same model, but for the target catalyst

$$y_i = \beta_{t0} + \sum_{j=1}^{p} \beta_{tj} x_{ij} + \epsilon_i \tag{4}$$

with $\epsilon_i \sim \mathcal{N}(0, {\sigma_t}^2)$ and for which the available training data set is of a smaller size n_t.

3.1.1 A parametric approach

The first transfer method for the linear model is a parametric approach inspired from Bouveyron and Jacques 2010, in which some regression parameters are kept unchanged for the target model ($\beta_{sj} = \beta_{tj}$ for some j), considering the influence doesn't change between both models, and then learn only other parameters. The standard deviation parameter σ_t is assumed to be equal to σ_s.

If $\mathcal{M}$ is the set of index of parameters to be modified, then $\beta_{tj} = \beta_{sj}$ for $j \in \{1, \ldots, p\} \backslash \mathcal{M}$ and $\beta_{tj} = \lambda_j \beta_{sj}$ for $j \in \mathcal{M}$. Then, only a reduced number of parameters have to be estimated for the target model. The challenge with this approach is how to chose $\mathcal{M}$, in particular the number of parameters and which ones. In the first step, we will assume $\#\mathcal{M}$ is known and we will select the best parameters by leave-one-out cross validation (LOOCV) on the n_t target points. When $\#\mathcal{M} = 1$, the best parameter is the one minimizing the RMSE by LOOCV. For $\#\mathcal{M} > 1$, we decided not to test all possible combinations

and rather to identify successively the next best parameter to modify. The final model is obtained by choosing the values of the $\#\mathcal{M}$ parameters minimizing the RMSE by LOOCV.

The results are presented in the left panel of Figure 3 for $\#\mathcal{M} \in \{1, 3, 8, 13\}$. With this approach, a performing model can be fitted with less points than if a totally new model is learned (Figure 3, left). For example, with 10 observations and modifying 1 or 3 parameters, RMSE is smaller than the objective of 0.005. In contrast, to achieve this result by learning the model from scratch (on all parameters), 30 observations are needed. Changing a small number of parameters, like one or two, is more efficient for small training sets but worse when a lot of data is available where a plateau is quickly reached. When $\#\mathcal{M}$ is higher, the number of points needed to get a satisfactory score is higher but the prediction accuracy keeps improving. Another remark is that for a given $\#\mathcal{M}$, the selected parameters change with changing sample size. In other words, we are not able to define the best parameters to transform with a few points.

The next step would be to decide on the optimal size of $\mathcal{M}$. Our idea is to select, among all the values of $\#\mathcal{M}$, the one with the best RMSE (LOOCV) for each n_t. As shown in Figure 3 (right), this approach does not give the lowest RMSE on the test set and is only marginally better than the model learned from scratch. So far, we did not find an effective method for determining the parameters and the number of parameters to be modified. For this reason, Bayesian inference is explored.

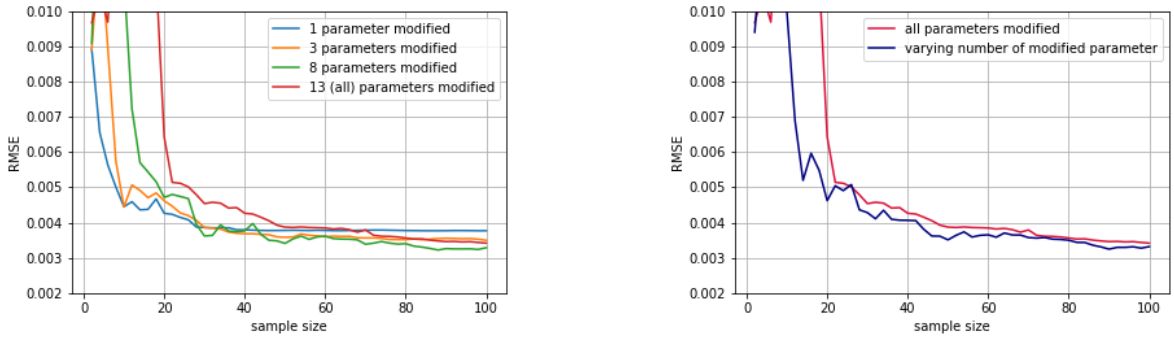

Figure 3: Graphs showing the change of RMSE according to n_t. On the left, the parameters to modify were chosen by cross validation. On the right, cross validation is also used to determine the number of parameters to be modified. The y-axis has been cut for better readability, as very high scores do not interest us.

3.1.2 A Bayesian approach

In this section, a Bayesian approach is used to learn parameters for the target linear model. The linear model for the target is:

$$\boldsymbol{y} = \boldsymbol{\beta}_t \boldsymbol{X} + \boldsymbol{\epsilon}_i$$

where $\boldsymbol{\beta}_t = (\beta_{t0}, \beta_{t1}, \ldots, \beta_{tp})^T$ is a random variable of prior density $\pi(\boldsymbol{\beta}_t)$, $\boldsymbol{X}$ is the design matrix into which a first unity column has been added. The Bayes Theorem gives that the posterior of $\boldsymbol{\beta}_t$ is

$$\pi(\boldsymbol{\beta}_t|\boldsymbol{y}_t, \boldsymbol{X}_t) \quad = \quad \frac{\pi(\boldsymbol{\beta}_t) f(\boldsymbol{y}_t|\boldsymbol{\beta}_t, \boldsymbol{X}_t)}{f(\boldsymbol{y}_t|\boldsymbol{X}_t)},$$

where $(\boldsymbol{y}_t, \boldsymbol{X}_t)$ are the target observations.

Different prior distributions $\pi(\boldsymbol{\beta}_t)$ are considered. The first one is the well known Zellner's prior (Zellner 1986), also known as g-prior, for the parameters $\boldsymbol{\beta}_t$:

$$\pi(\boldsymbol{\beta}_t) = \mathcal{N}(\widehat{\boldsymbol{\beta}}_s, g\sigma_t^2(\boldsymbol{X}_t^T \boldsymbol{X}_t)^{-1}),$$

where $\widehat{\boldsymbol{\beta}}_s$ is the maximum likelihood estimator (MLE) learned on the source data. By using such a prior, only the mean of the prior distribution depends on the source data. The structure of the prior covariance of $\boldsymbol{\beta}_t$ depends on the target data, and a scalar parameter g allows the impact of the prior distribution to be tuned. Notice that the posterior's mean using such a prior is a weighted average between the MLE for source data and the MLE for target data: $\widehat{\boldsymbol{\beta}}_t = \frac{1}{g+1}(g\widehat{\boldsymbol{\beta}}_{MLE,t} + \widehat{\boldsymbol{\beta}}_s)$.

The results are presented in Figure 4. With this prior, the results are not satisfactory irrespective of the value of g. For a target sample size lower than 15, the linear model estimated without transfer (directly from the target data) has a lower RMSE (blue line on Figure 4). For a larger sample size, the results are better for the transferred linear model, but the desired RMSE score of 0.005 is not reached.

With this Zellner's prior, the source dataset acts only on the mean of the prior distribution. One idea to improve the results is to increase the information transferred by also acting on prior covariance. Indeed, when n_t is small, the covariance structure cannot be well estimated. We consequently proposed the following prior:

$$\pi(\boldsymbol{\beta}_t) = \mathcal{N}(\widehat{\boldsymbol{\beta}}_s, g\sigma_s^2(\boldsymbol{X}_s^T \boldsymbol{X}_s)^{-1}),$$

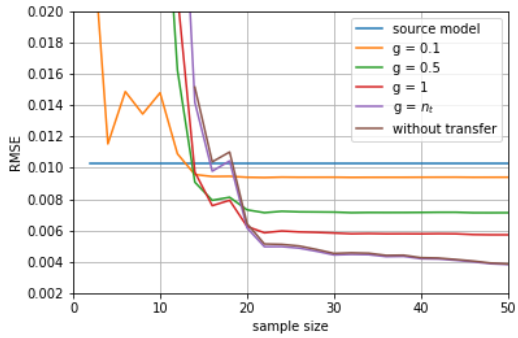

Figure 4: Comparison between an estimation of $\boldsymbol{\beta}_t$ with a Bayesian approach and a g-prior for different values of g, and a model learned without any prior.

in which the covariance structure is now estimated from the source data. The mean of the corresponding posterior distribution is equal to:

$$\widehat{\boldsymbol{\beta}}_t = (\boldsymbol{X}_t^T \boldsymbol{X}_t + \sigma_t^2 g^{-1} \boldsymbol{\Sigma}_s^{-1})^{-1} (\boldsymbol{X}_t^T \boldsymbol{y}_t + \sigma_t^2 g^{-1} \boldsymbol{\Sigma}_s^{-1} \widehat{\boldsymbol{\beta}}_s),$$

with $\boldsymbol{\Sigma}_s = \sigma_s^2 (\boldsymbol{X}_s^T \boldsymbol{X}_s)^{-1}$. We notice that when $g \to \infty$ the posterior mean tends to the MLE learned on target observation. When $g \to 0$ the posterior mean tends to the prior mean. The RMSE scores are computed for different values of g in Figure 5 (left panel). With this prior, we pass under the threshold of 0.005 for the RMSE score for all g values greater than 1. Furthermore, g values between 100 and 1,000 lead to better results than those obtained when estimated the model directly from the target data without transfer. However, g value as to be chosen and we propose a pragmatic empirical strategy.

Since data are normalized, elements of parameter $\boldsymbol{\beta_t}$ take values close to $[-1, 1]$. In order to allow a parameter to change in this range, the prior variance should be close to 1. A suitable g value should be around of the inverse of the average of the diagonal elements of $\boldsymbol{\Sigma}_s$. Following this strategy leads to a g value approximately equal to 800 in our experiment. The corresponding results are shown in the right panel of Figure 5.

With this approach, the RMSE scores are always lower than those obtained without transfer. Moreover, only 5 target points are needed to reach the industrial constraint of a RMSE lower than 0.005. In comparison, estimating the target model without transfer learning will need at least 50 observations.

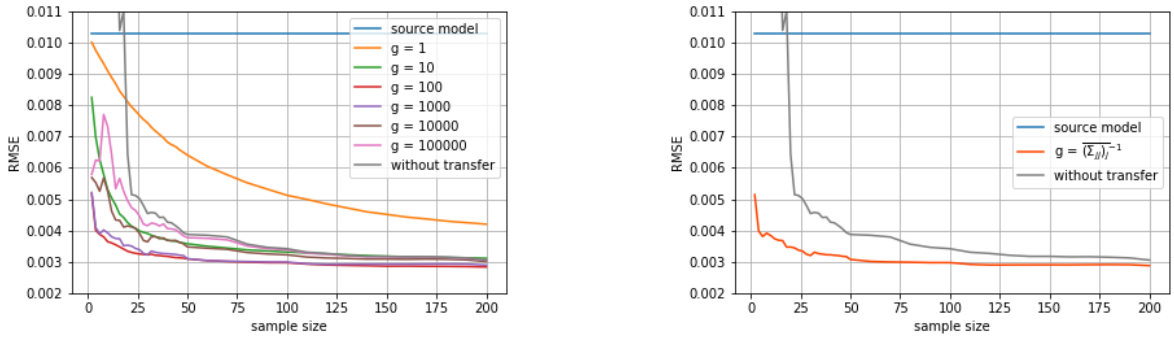

Figure 5: Impact of g on the Bayesian linear model. The graph shows the change in RMSE for different values of g according to n_t.

3.2 Transfer learning for the Kriging Model

Following these encouraging results in the Bayesian transfer of linear model (only 5 points are needed for the target data), we will now focus on transferring the kriging model using a Bayesian approach. The kriging model (2) is composed of two parts, a trend part and a Gaussian Process part:

$$y_i = \Sigma_{j=0}^{d} \beta_j x_{ij} + \delta(\boldsymbol{x}_i).$$

Inspired from the Bayesian transfer for the linear model, a first Bayesian kriging model is considered with a prior on the trend part only (Section 3.2.1). In a second step, we additionally consider a prior distribution for the covariance of the Gaussian process (Section 3.2.2).

3.2.1 A Bayesian transfer on trend parameters only

Here, the covariance function C is assumed to be known and identical to the covariance function of the source model. Consequently, its parameters $\boldsymbol{\theta}_t, \sigma_t$ and τ_t are equal to $\hat{\boldsymbol{\theta}}_s, \hat{\sigma}_s$ and $\hat{\tau}_s$, respectively. The covariance of the Gaussian Process used for prediction is calculated only for observations from the target dataset:
$\boldsymbol{C}_t = (C(|\boldsymbol{x}_{ti} - \boldsymbol{x}_{ti'}|))_{1 \leq i, i' \leq n_t}$.

As in the Bayesian transfer of the linear model, a Gaussian prior distribution is considered for $\boldsymbol{\beta}_t$, with mean $\widehat{\boldsymbol{\beta}}_s$ and variance $\boldsymbol{\Sigma}_s$ modified by a factor g:

$$\pi(\boldsymbol{\beta}_t) \quad = \quad \mathcal{N}(\widehat{\boldsymbol{\beta}}_s, g\boldsymbol{\Sigma}_s),$$

where $\boldsymbol{\Sigma}_s = (\boldsymbol{X}_s^T \boldsymbol{C}_s^{-1} \boldsymbol{X}_s)^{-1}$, $\boldsymbol{C}_s = (\widehat{\sigma}_s^2 R_{\widehat{\boldsymbol{\theta}}_s, \widehat{\tau}_s}(|\boldsymbol{x}_{si} - \boldsymbol{x}_{si'}|))_{1 \leq i, i' \leq n_s}$.

This prior leads to the following posterior distribution for $\boldsymbol{\beta}_t$ (as used in Helbert, Dupuy, and Carraro 2009):

$$\pi(\boldsymbol{\beta}_t | \boldsymbol{y}, \boldsymbol{X}) = \mathcal{N}(\widehat{\boldsymbol{\beta}}_s + g\boldsymbol{\Sigma}_s \boldsymbol{X}^T (g\boldsymbol{X}\boldsymbol{\Sigma}_s \boldsymbol{X}^T + \boldsymbol{C}_s)^{-1} (\boldsymbol{y} - \boldsymbol{X}\widehat{\boldsymbol{\beta}}_s),$$

$$g\boldsymbol{\Sigma}_s - g\boldsymbol{\Sigma}_s \boldsymbol{X}^T (g\boldsymbol{X}\boldsymbol{\Sigma}_s \boldsymbol{X}^T + \boldsymbol{C}_s)^{-1} g\boldsymbol{X}\boldsymbol{\Sigma}_s)$$

Figure 6 presents the corresponding RMSE scores, for different values of g. Irrespective of the value of g, the transferred kriging models overperform the kriging model estimated on the target data only.

Similar to the linear model, g is chosen to be equal to the inverse of the average of the diagonal of $\boldsymbol{\Sigma}_s$ ($g \simeq 200$). We notice that other choices for g could lead to even better results, but the choice of g is difficult when using a small target sample, and we advise on considering our heuristic value for g rather than trying to tune it by cross-validation.

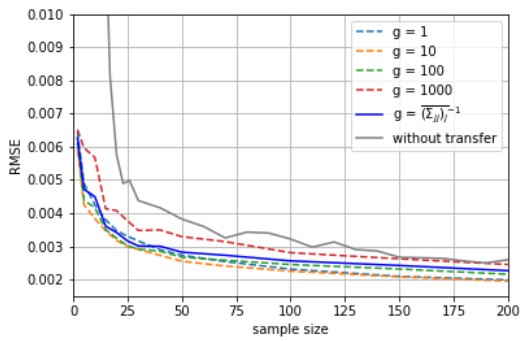

Figure 6: Impact of g on the Bayesian kriging model. The graph shows the change in RMSE for different values of g according to n_t.

With Bayesian transfer on the trend, 5 target observations are enough to reach a RMSE score of 0.005 instead of about 25 for a kriging model learned on the target data only. The best RMSE score reached with the transferred linear model with 50 points (0.003, Figure 5), is obtained with only 25 points only and is outperformed when increasing the number of points. In addition to reducing the number of points required to learn a model that meets the performance criteria, the Bayesian transfer of the trend of a kriging model also improves the performance of the model for a larger number of points.

3.2.2 A fully Bayesian approach

In this section, we add a prior distribution on the covariance function parameters $\boldsymbol{\theta}_t, \sigma_t$ and τ_t.

Prior distribution The choice of prior for $\boldsymbol{\beta}_t$ is still a Gaussian distribution, and the prior for the other parameters are Gamma distributions. The Gamma distributions are such that the mean and variance are those of the source parameters:

$$\begin{aligned}
\pi(\boldsymbol{\beta}_t, \boldsymbol{\theta}_t, \sigma_t, \tau_t) &= \pi(\boldsymbol{\beta}_t | \boldsymbol{\theta}_t, \sigma_t, \tau_t)\pi(\boldsymbol{\theta}_t, \sigma_t, \tau_t), \\
\pi(\boldsymbol{\beta}_t | \boldsymbol{\theta}_t, \sigma_t, \tau_t) &= \mathcal{N}(\widehat{\boldsymbol{\beta}}_s, g\boldsymbol{\Sigma}_s), \\
\pi(\boldsymbol{\theta}_t, \sigma_t, \tau_t) &= \prod_j \pi(\theta_{tj}) \times \pi(\sigma_t) \times \pi(\tau_t), \\
\pi(\theta_{tj}) = \mathcal{G}(\cdot,\cdot) & \quad \text{with hyperparameter s.t. } \mathbb{E}(\theta_{tj}) = \hat{\theta}_{sj} \text{ and } Var(\theta_{tj}) = Var(\hat{\theta}_{sj}), \\
\pi(\sigma_t) = \mathcal{G}(\cdot,\cdot) & \quad \text{with hyperparameter s.t. } \mathbb{E}(\sigma_t) = \hat{\sigma}_s \text{ and } Var(\sigma_t) = Var(\hat{\sigma}_s), \\
\pi(\tau_t) = \mathcal{G}(\cdot,\cdot) & \quad \text{with hyperparameter s.t. } \mathbb{E}(\tau_t) = \hat{\tau}_s \text{ and } Var(\tau_t) = Var(\hat{\tau}_s),
\end{aligned}$$

With these priors, no closed form exists for the posterior distribution and a MCMC algorithm is used to estimate the posterior distributions. Parameter g and matrix $\boldsymbol{\Sigma}_s$ are chosen in the same manner as in the previous section.

Markov Chain Monte Carlo (MCMC) algorithm The goal is to approximate the posterior distribution of $(\boldsymbol{\beta}_t, \boldsymbol{\theta}_t, \sigma_t, \tau_t)$. The MCMC algorithm is an iterative algorithm that generates a Markov chain whose stationary distribution is the desired posterior. At each iteration, new values for the parameters are generated and we note the values of parameters $\boldsymbol{\beta}_t^{(q)}, \boldsymbol{\theta}_t^{(q)}, \sigma_t^{(q)}, \tau_t^{(q)}$ at iteration (q). The MCMC algorithm used is a Metropolis Hastings within Gibbs algorithm (Tierney 1994), which is detailed below.

At each iteration, the parameters are updated sequentially. For any scalar parameter ρ, a new value ρ^{new} is proposed such that $\rho^{new} = \rho^{(q)} + r$ where r is randomly drawn according to a centered Gaussian distribution of variance specific to each parameter. The idea of Metropolis-Hastings is to accept ρ^{new} as the new value if its posterior is better than that of $\rho^{(q)}$, and randomly otherwise. We compute the ratios of the posterior as the ratio of the product between the likelihood and the prior, with all other parameters remaining unchanged and $\rho^{(q+1)}$ chosen to be ρ^{new} with probability $min(1, ratio)$ and $\rho^{(q)}$ otherwise. The variance of r is chosen empirically so that the acceptance rates are between 20% and 60% for each parameter during all the iterations. Consequently, there is a need to run the

algorithm a few times in order to tune these standard deviations. In our application, the following standard deviation was considered: $\xi_{\beta_{tj}} = 0.2$, $\xi_{\theta_{tj}} = 1$ and $\xi_{\sigma_t} = \xi_{\tau_t} = 0.03$.

The inputs of the algorithm are the target data $\boldsymbol{X}_t, \boldsymbol{y}_t$ and the prior distribution $\pi(\boldsymbol{\beta}_t, \boldsymbol{\theta}_t, \sigma_t, \tau_t)$ defined above. Let $\mathcal{N}(\cdot; \boldsymbol{\mu}, \boldsymbol{\Sigma})$ be the multivariate Gaussian probability density function (p.d.f) of parameter $(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. Let $\mathcal{G}^{(\theta_{tj})}(\cdot), \mathcal{G}^{(\sigma_t)}(\cdot), \mathcal{G}^{(\tau_t)}(\cdot)$ be the Gamma p.d.f. used as prior $\pi(\theta_{tj}), \pi(\sigma_t), \pi(\tau_t)$ for $j \in 1, \ldots, p$. Let $\boldsymbol{\rho}^{(q)}$ be $(\theta_{t1}^{(q)}, \ldots, \theta_{tp}^{(q)}, \sigma_t^{(q)}, \tau_t^{(q)})$ and $C(h_1, ..., h_p; \boldsymbol{\rho}^{(q)}) = \sigma_t^{(q)2} R_{\boldsymbol{\theta}_t^{(q)}, \tau_t^{(q)}}(h_1, ..., h_p)$.

Initialization of parameters, $(\boldsymbol{\beta}_t^{(0)}, \boldsymbol{\theta}_t^{(0)}, \sigma_t^{(0)}, \tau_t^{(0)})$, is performed using the mean of the prior distribution. The algorithm for updating the parameters is slightly different for $\boldsymbol{\beta}_t$ that follows a Gaussian distribution and $\{\theta_{tj}\}_j, \sigma_t$ and τ_t that follow a Gamma distribution. The update of $\boldsymbol{\beta}_t^{(q)}$ is detailed in Algorithm 1. The update of $\{\theta_{tj}\}_j, \sigma_t$ and τ_t is detailed in Algorithm 2. The main difference is that, for the $\boldsymbol{\beta}_t$ case, the covariance of the Gaussian Process is not a function of the parameters and can be computed outside of the $\boldsymbol{\beta}_t$ update loop as $\boldsymbol{C} = \sigma_t^{(q)2} R_{\boldsymbol{\theta}_t^{(q)}, \tau_t^{(q)}}(|\boldsymbol{x}_{ti} - \boldsymbol{x}_{ti'}|))_{1 \leq i, i' \leq n_t}$ with the values of the previous iteration $\boldsymbol{\theta}_t^{(q)}, \sigma_t^{(q)}, \tau_t^{(q)}$.

Algorithm 1 Update of the parameter $\{\beta_{tj}\}_j$ at iteration $(q+1)$.

$\boldsymbol{\beta}_t^{cur} \leftarrow \boldsymbol{\beta}_t^{(q)}$
for $j = 0$ to p **do**
  $\boldsymbol{\beta}_t^{new} \leftarrow \boldsymbol{\beta}_t^{cur}$
  $\beta_{tj}^{new} \leftarrow \beta_{tj}^{(q)} + r$ with $r \sim \mathcal{N}(0, \xi_{\beta_{tj}})$
  compute $ratio = \frac{\mathcal{N}(\boldsymbol{y}_t; \boldsymbol{\beta}_t^{new} \boldsymbol{X}_t, \boldsymbol{C})\, \mathcal{N}(\boldsymbol{\beta}_t^{new}; \widehat{\boldsymbol{\beta}}_s, g\boldsymbol{\Sigma}_s)}{\mathcal{N}(\boldsymbol{y}_t; \boldsymbol{\beta}_t^{cur} \boldsymbol{X}_t, \boldsymbol{C})\, \mathcal{N}(\boldsymbol{\beta}_t^{cur}; \widehat{\boldsymbol{\beta}}_s, g\boldsymbol{\Sigma}_s)}$;
  generate $u \sim \mathcal{U}([0, 1])$
  if $u < min(1, ratio)$ **then**
    $\boldsymbol{\beta}_t^{cur} \leftarrow \boldsymbol{\beta}_t^{new}$
  end if
end for
$\boldsymbol{\beta}_t^{(q+1)} \leftarrow \boldsymbol{\beta}_t^{cur}$

The number of iterations of the MCMC algorithm must be large enough so that the Markov chain converges to its stationary distribution. In practice, this number is fixed by observing the evolution of the Markov chain over the iterations and must have reached its stationary distribution for thousands of observations. Since the generated Markov chain needs a given time to reach its stationary distribution, the first iterations are dropped. The

Algorithm 2 Update of the parameters $\{\theta_{tj}\}_j, \sigma_t$ and τ_t at iteration $(q+1)$.

$\boldsymbol{\rho}^{cur} \leftarrow \boldsymbol{\rho}^{(q)}$
for $\rho^{(q)} \in \boldsymbol{\rho}^{(q)}$ **do**
  $\boldsymbol{\rho}^{new} \leftarrow \boldsymbol{\rho}^{cur}$
  $\rho^{new} \leftarrow \rho^{(q)} + r$ with $r \sim \mathcal{N}(0, \xi_\rho)$
  compute
    $\boldsymbol{C}^{new} = (C(|\boldsymbol{x}_{ti} - \boldsymbol{x}_{ti'}|; \boldsymbol{\rho}^{new}))_{1 \leq i,i' \leq n_t}$
    $\boldsymbol{C}^{cur} = (C(|\boldsymbol{x}_{ti} - \boldsymbol{x}_{ti'}|; \boldsymbol{\rho}^{cur}))_{1 \leq i,i' \leq n_t}$
  compute $ratio = \frac{\mathcal{N}(\boldsymbol{y}_t; \boldsymbol{\beta}_t^{(q+1)} \boldsymbol{X}_t, \boldsymbol{C}^{new})\, \mathcal{G}^{(\rho)}(\rho^{new})}{\mathcal{N}(\boldsymbol{y}_t; \boldsymbol{\beta}_t^{(q+1)} \boldsymbol{X}_t, \boldsymbol{C}^{cur})\, \mathcal{G}^{(\rho)}(\rho^{cur})}$
  generate $u \sim \mathcal{U}([0,1])$
  if $u < min(1, ratio)$ **then**
    $\boldsymbol{\rho}^{cur} \leftarrow \boldsymbol{\rho}^{new}$
  end if
end for
$\boldsymbol{\rho}^{(q+1)} \leftarrow \boldsymbol{\rho}^{cur}$

size of this burn-in period depends on how far $(\boldsymbol{\beta}_t^{(0)}, \boldsymbol{\theta}_t^{(0)}, \sigma_t^{(0)}, \tau_t^{(0)})$ is from the mean of the posterior distribution. It is also fixed by observing the evolution of the Markov chain. In our application, 5,000 MCMC iterations with a burn-in period of 500 is considered. Finally, estimation of $(\widehat{\boldsymbol{\beta}}_t, \widehat{\boldsymbol{\theta}}_t, \widehat{\sigma}_t, \widehat{\tau}_t)$ is obtained by computing the empirical mean of the marginal posterior distribution.

Experimental results The results are presented in Figure 7. For a reduced number of target observations, and up to about 50 points, adding a prior on the covariance function parameter does not improve the results. But when the number of points is greater than 50, it slightly improves the results compared to the Bayesian kriging on trend parameters. This allows us to obtain particularly good predictions.

3.2.3 Conclusion for Bayesian transfer of the kriging model

To conclude, transfer learning for the kriging method is very efficient. To obtain an RMSE score of 0.004, one needs about 5 points instead of about 25 without transfer learning. For scores of 0.004 and 0.003, it is 12 versus 45 and 25 versus 110, respectively. An RMSE score of 0.0019 is obtained with 200 points, which is better than the model learned from scratch and probably not possible without transfer learning.

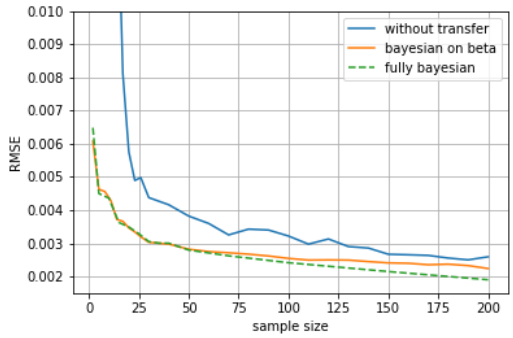

Figure 7: Comparison of the different kriging approaches tested.

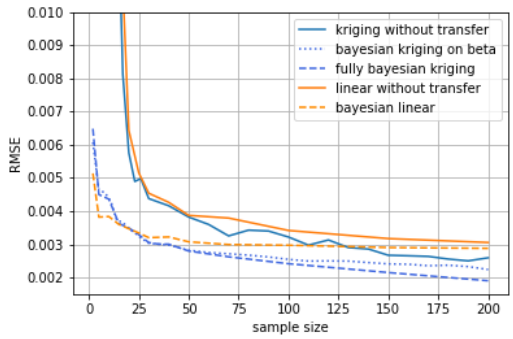

Figure 8: Comparison of Linear Model and Kriging results.

4 Conclusion and future work

The objective was to build an efficient predictive model for a new catalyst with few new data. To do this, we used a second dataset from an old catalyst for which many observations were available. Different models were tested on this second dataset in order to build an efficient predictive model. Transfer learning approaches were then tested on the linear and kriging models to build the new model for the new catalyst. The Bayesian approach carried out for the two models, the kriging and linear model, gives us a real improvement in terms of the number of observations needed to learn a performing model (Figure 8). With only 5 new points, the linear model transferred with the Bayesian approach give us an RMSE score that requires about 50 points for a model learned without knowledge of the source

dataset. This represents a cost of 50k euros instead of 500k euros. For the kriging model transferred with the Bayesian approach, 12 new points are required. For a small size of n_t, lower than 15, the transferred linear model is slightly better than the transferred kriging model. With more target points, however, the transferred kriging model improves and the gap widens when the number of points becomes larger. Our score objective is reached with 5 points and Bayesian linear model, but if one wants to get an even better score, kriging provides a solution. Moreover, even when the number of new observations is relatively large, the use of this method improves the prediction performance of the model.

The Bayesian transfer approach is therefore recommended, regardless of the number of points available. If a reduced number of points is available, less than 15, it is advisable to use a simple model such as the linear model for Bayesian transfer. The RMSE score is below the target of 0.005 with only 3 new points and quickly goes below 0.004. If more points are available, it would then be advisable to use the full Bayesian kriging model, which would lead to better results.

We have applied this method for the case of diesel density prediction, but it can be applied for any type of problem for which the kriging or linear models are efficient, and therefore for any type of product quality. A possibility is to extend this Bayesian approach to kinetic models, consisting of solving a differential equation system, which are widely used to simulate the hydrocracking process (Ancheyta, Sánchez, and Rodríguez 2005; Becker, Serrand, et al. 2017; N. Y. P. Cao et al. 2020). Further work will also focus on the design of experiments. In this paper, the data points from the target set were selected randomly. The results can certainly be improved by choosing the points to be measured to build the new model. The Kennard-Stone's algorithm (Kennard and L. Stone 1969) is a possibility that will be explored. Other approaches such as D-optimality or A-optimality have proven their worth in experimental design (Celse, J. J. D. Costa, and V. Costa 2016; De Aguiar et al. 1995; Nikolov, Singh, and Tantipongpipat 2019) and will be tested.

References

Ancheyta, J., S. Sánchez, and M. A. Rodríguez (2005). "Kinetic modeling of hydrocracking of heavy oil fractions: A review". In: *Catalysis Today*, pp. 76–92.

Argyriou, A., T. Evgeniou, and M. Pontil (2007). "Multi-task feature learning". In: *Advances in neural information processing systems*, pp. 41–48.

Argyriou, A., M. Pontil, et al. (2008). "A spectral regularization framework for multi-task structure learning". In: *Advances in neural information processing systems*, pp. 25–32.

Becker, P. J., B. Celse, et al. (2016). “A continuous lumping model for hydrocracking on a zeolite catalysts: model development and parameter identification”. In: *Fuel*, pp. 73–82.

Becker, P. J., N. Serrand, et al. (2017). “A single events microkinetic model for hydrocracking of vacuum gas oil”. In: *Computers Chemical Engineering*, pp. 70–79.

Bouveyron, C. and J. Jacques (2010). “Adaptive linear models for regression: improving prediction when population has changed”. In: *Pattern Recognition Letters*, pp. 2237–2247.

Breunig, M.M. et al. (2000). “LOF: identifying density-based local outliers”. In: *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pp. 93–104.

Cao, B. et al. (2010). “Adaptive Transfer Learning”. In: *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence*, pp. 407–412.

Cao, N. Y. P. et al. (2020). “Accelerating Kinetic Parameter Identification by Extracting Information from Transient Data: A Hydroprocessing Study Case”. In: *Catalysts*, p. 361.

Celse, B., J. J. D. Costa, and V. Costa (2016). “Experimental Design in Nonlinear Case Applied to Hydrocracking Model: How Many Points Do We Need and Which Ones?” In: *International Journal of Chemical Kinetics*, pp. 660–670.

Cressie, N. (1988). “Spatial prediction and ordinary kriging”. In: *Mathematical geology*, pp. 405–421.

— (1990). “The origins of kriging”. In: *Mathematical geology*, pp. 239–252.

Dai, W. et al. (2007). “Boosting for Transfer Learning”. In: *Proceedings of the 24th international conference on Machine learning*, pp. 193–200.

De Aguiar, P. F. et al. (1995). “D-optimal designs”. In: *Chemometrics and intelligent laboratory systems* 30, pp. 199–210.

Evgeniou, T. and M. Pontil (2004). “Regularized multi-task learning”. In: *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 109–117.

Helbert, C., D. Dupuy, and L. Carraro (2009). “Assessment of uncertainty in computer experiments from Universal to Bayesian Kriging”. In: *Applied Stochastic Models in Business and Industry*, pp. 99–113.

Kennard, R.W. and L.A. Stone (1969). “Computer aided design of experiments”. In: *Technometrics*, pp. 137–148.

Launay, T., A. Philippe, and S. Lamarche (2015). “Construction of an informative hierarchical prior for a small sample with the help of historical data and application to electricity load forecasting”. In: *Test*, pp. 361–385.

Nikolov, A., M. Singh, and U. T. Tantipongpipat (2019). "Proportional volume sampling and approximation algorithms for A-optimal design". In: *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*. Society for Industrial and Applied Mathematics, pp. 1369–1386.

Pan, S. and Q. Yang (2010). "A Survey on Transfer Learning". In: *IEEE Transactions on Knowledge and Data Engineering*, pp. 1345–1359.

Pardoe, D. and P. Stone (2010). "Boosting for Regression Transfer". In: *Proceedings of the 27th international conference on Machine learning*, pp. 863–870.

Pedregosa, F. et al. (2011). "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12, pp. 2825–2830.

Raina, R., A. Y. Ng, and D. Koller (2006). "Constructing informative priors using transfer learning". In: *Proceedings of the 23rd international conference on Machine learning*, pp. 713–720.

Roustant, O., D. Ginsbourger, and Y. Deville (2012). "Dicekriging, Diceoptim: Two R packages for the analysis of computer experiments by kriging-based metamodelling and optimization". In: *Journal of Statistical Software*, 54p.

Salaken, S. M. et al. (2019). "Seeded transfer learning for regression problems with deep learning". In: *Expert Systems with Applications*, pp. 565–577.

Segev, N. et al. (2016). "Learn on source, refine on target: a model transfer learning framework with random forests". In: *IEEE Transactions on pattern analysis and machine intelligence*, pp. 1811–1824.

Stein, M. L. (2012). *Interpolation of spatial data: some theory for kriging*. Springer Science & Business Media.

Tierney, L. (1994). "Markov chains for exploring posterior distributions". In: *the Annals of Statistics*, pp. 1701–1728.

Tsung, F. et al. (2018). "Statistical transfer learning: A review and some extensions to statistical process control". In: *Quality Engineering*, pp. 115–128.

Zellner, A. (1986). "On assessing prior distributions and Bayesian regression analysis with g-prior distributions". In: *Goel, P. and Zellner, A., Eds., Bayesian Inference and Decision Techniques: Essays in Honor of Bruno de Finetti, Elsevier Science Publishers*, pp. 233–243.